

**Azure Monitor Implementation**

**Version Control**

All information given under this version over rides all and any kind of offers, assumptions, deliverables and contracts, given under any previous version. All previous versions of the subject proposal stand null and void.

**Restrictions**

The data contained in this document shall not be disclosed and shall not be duplicated, used, or disclosed in whole or in part for any purpose. If a contract is awarded to chosen parties as a result of or in connection with the submission of this data, the client or prospective client shall have the right to duplicate, use, or disclose this data to the extent provided in the contract. This restriction does not limit the client’s or prospective client’s right to use the information contained in the data if it is obtained from another source without restriction. The data subject to this restriction is contained in all marked sheets.

Table of Contents

[1 DOCUMENT OBJECTIVE 2](#_Toc168915946)

[2 SERVICE DESCRIPTION – AZURE MONITOR OVERVIEW 3](#_Toc168915947)

[2.1 What is Azure Monitor? 3](#_Toc168915948)

[2.2 What do we want to achieve with this implementation? 3](#_Toc168915949)

[3 HIGH-LEVEL OVERVIEW OF AZURE MONITOR 3](#_Toc168915950)

[3.1 Resource Group 3](#_Toc168915951)

[3.2 Log Analytics Workspace - Why do we need LAW? 3](#_Toc168915952)

[3.3 Data Collection Rule - Why do we need DCR? 3](#_Toc168915953)

[3.4 Alerts - Why do we need Alerts? 3](#_Toc168915954)

[3.5 Action Group 4](#_Toc168915955)

[4 AZURE DEVOPS SYSTEM OVERVIEW 4](#_Toc168915956)

[4.1 Create a New Organization & Project for a new customer. 4](#_Toc168915957)

[4.2 Create Azure Repos 5](#_Toc168915958)

[4.3 Create Release Pipelines 10](#_Toc168915959)

[4.4 Codes in Azure DevOps Organization 25](#_Toc168915960)

[v5. ALERTS TO BE CONFIGURED WITH TERRAFORM CODES IN AZURE DEVOPS. 191](#_Toc168915961)

[4.5 Platform Services, Virtual Machines & OS Metrices 191](#_Toc168915962)

[4.6 SQL Database Metrics 193](#_Toc168915963)

[4.7 PAAS Services 194](#_Toc168915964)

[4.7.3 Application Gateway Metrics 195](#_Toc168915967)

[4.7.4 Traffic Manager 195](#_Toc168915968)

[4.7.7 MySQL Database Metrics 197](#_Toc168915971)

[4.7.8 Postgre SQL DB 197](#_Toc168915972)

[4.7.9 Azure Data factory 198](#_Toc168915973)

[4.7.10 Azure Kubernetes Services 198](#_Toc168915974)

[5 TERRAFORM CODE DETAILS FOR ALL ALERTS - LOCATION 199](#_Toc168915975)

[6 CODE DEPLOYMENT STEPS VIA AZURE DEVOPS SYSTEM 199](#_Toc168915976)

[7 BEST PRACTICES 199](#_Toc168915977)

[7.1 Log Analytics Workspaces Best Practice 199](#_Toc168915978)

[7.2 Data Collection Rules 200](#_Toc168915979)

[7.3 VM Monitoring Best Practice 203](#_Toc168915980)

[8 IMPORTANT LINKS 205](#_Toc168915981)

[9 Known Error Database (KEDB) 205](#_Toc168915982)

[10 FAQ’s 206](#_Toc168915983)

[11 Limitations-Terraform and Azure 206](#_Toc168915984)

# DOCUMENT OBJECTIVE

The document is to provide detailed information, best practices and easy & simple step by step guidance with screen shots, to new joiners in the team, so that they can refer to this document and deploy use cases successfully. The document contains standard framework to implement Azure Monitor for a new customer onboarding.

# SERVICE DESCRIPTION – AZURE MONITOR OVERVIEW

## What is Azure Monitor?

Azure Native Service to monitor cloud services and workloads for performance and availability.

## What do we want to achieve with this implementation?

We want to implement and deploy Azure Monitor Services, such as Log Analytics Workspace, DCR and Alerts via Azure DevOps System, using Terraform codes.

# HIGH-LEVEL OVERVIEW OF AZURE MONITOR

Using Terraform codes, from Azure DevOps System, we will deploy Azure Monitor resources as following, through pipelines.

## Resource Group

We will need resource group to be created in customer environment, possible name as “RG-AM-Hub-Subscription”, “RG-AM-Dev-Subscription”, “RG-AM-Prod-Subscription”.

## Log Analytics Workspace - Why do we need LAW?

Log analytics workspace (LAW) is used to send and store the logs, it further analysis to edit and run log queries for the data collected from Azure monitor

## Data Collection Rule - Why do we need DCR?

Data collection rule (DCR) is a process of collecting data source from various resource types such as windows event log, IIS logs, linux syslog and custom text logs and send the data to respective LAW.

## Alerts - Why do we need Alerts?

Basically, alerts are proactive system notification in the event of any detection or failure in infrastructure operation to address the issue.

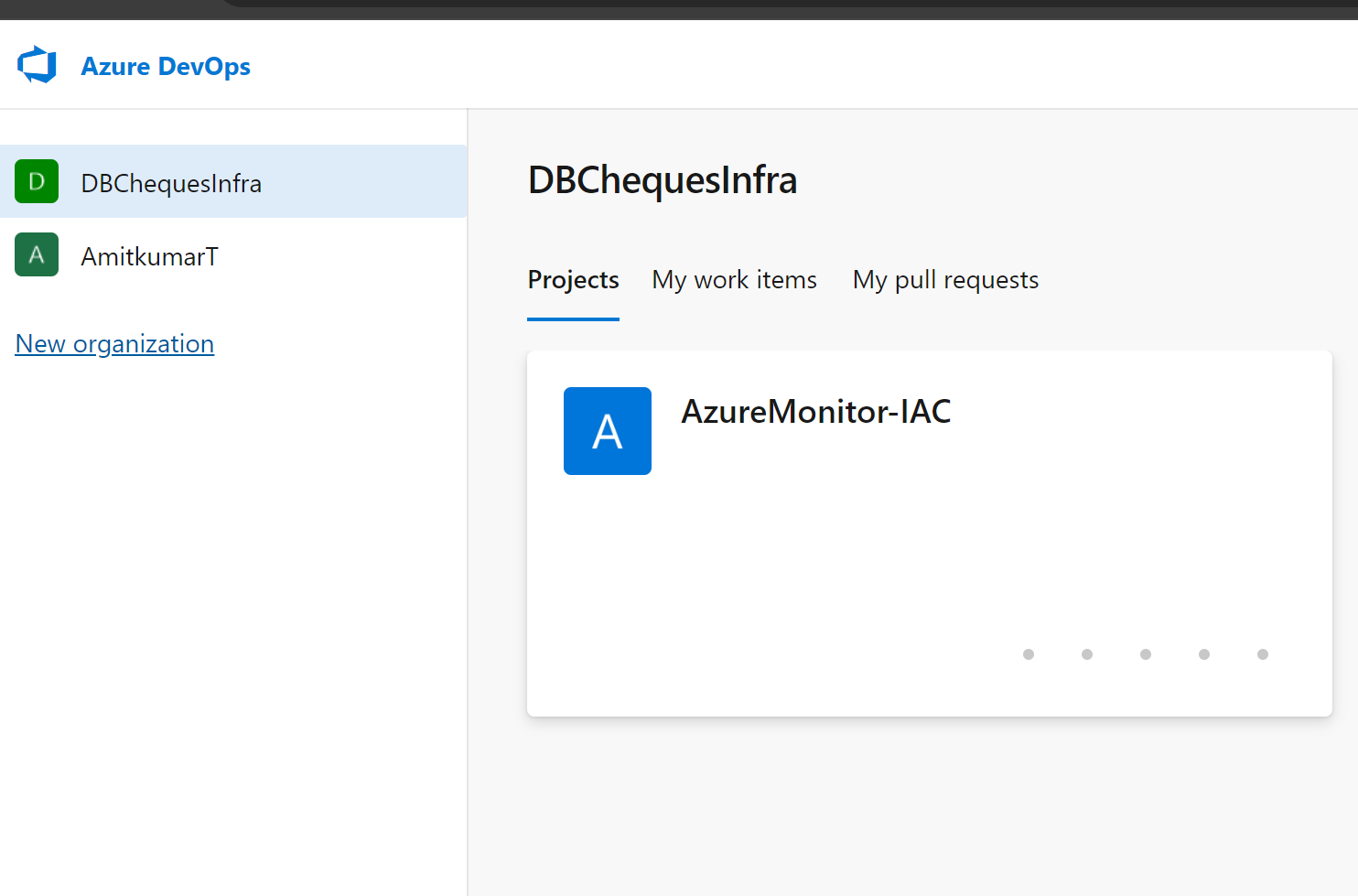
## Action Group

Action groups is configured for the **next steps**, post alarm is fired. So that, the alarms gets consumed either by human resources or by automation engine. It essentially notifies an alert once Azure monitor identifies a problem in infrastructure operation that correspond via email, message, voice call or any further automated actions.

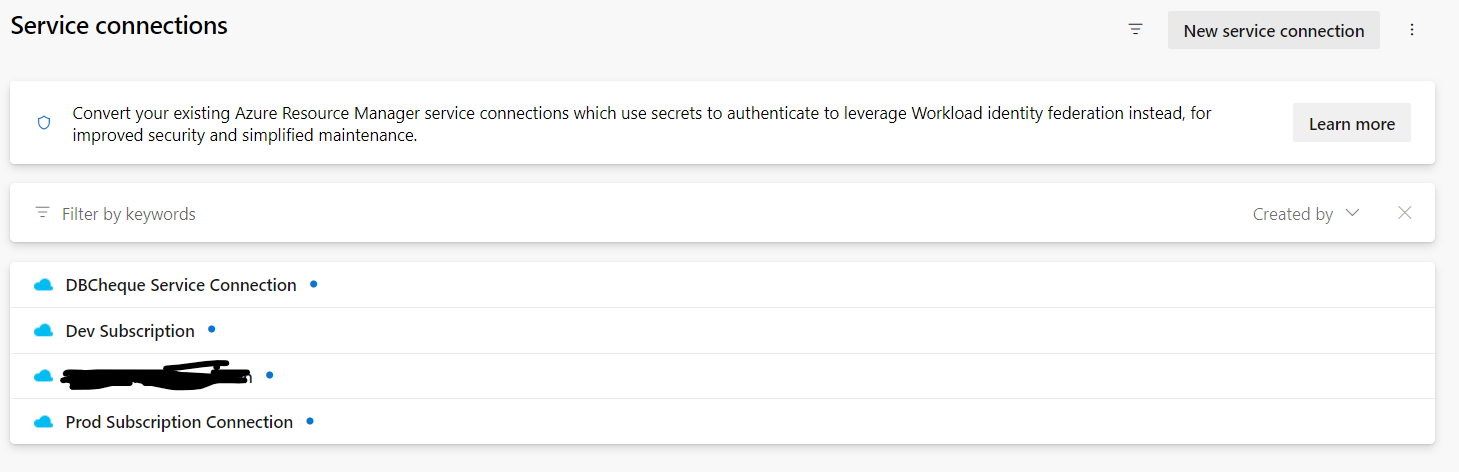
# AZURE DEVOPS SYSTEM OVERVIEW

## Create a New Organization & Project for a new customer.

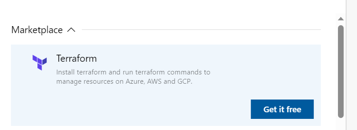
* Open a web browser and go to the Azure DevOps website **(**[**https://dev.azure.com/**](https://dev.azure.com/)**).**
* Sign in with your Azure DevOps account or create a new one if you don’t have an account.
* As shown in the below screenshot, “DBCHEQUES " organization is already created. Under this organization, project called “DBCHEQUES” is already present.



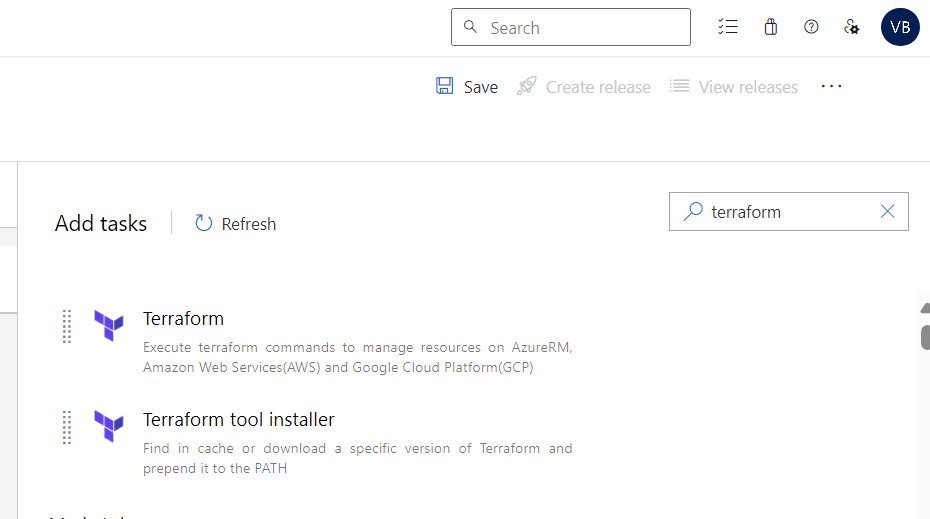
* As shown below, total 03 service connection is created for each 3 subscriptions individually.

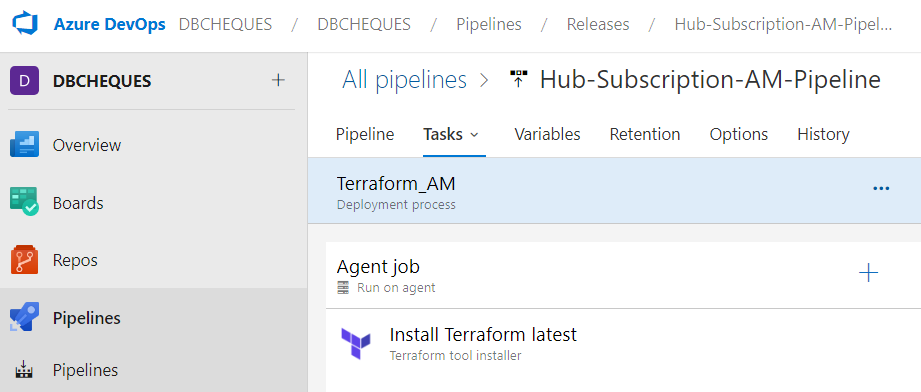


* Install terraform Extension
  + Click on “Get it free”.



* Select or create your organization.
* Here, the organization “DBCheques” already present as shown below:
* Under this org. create azure repos(files/folder) which includes terraform codes and powershell code for monitoring.

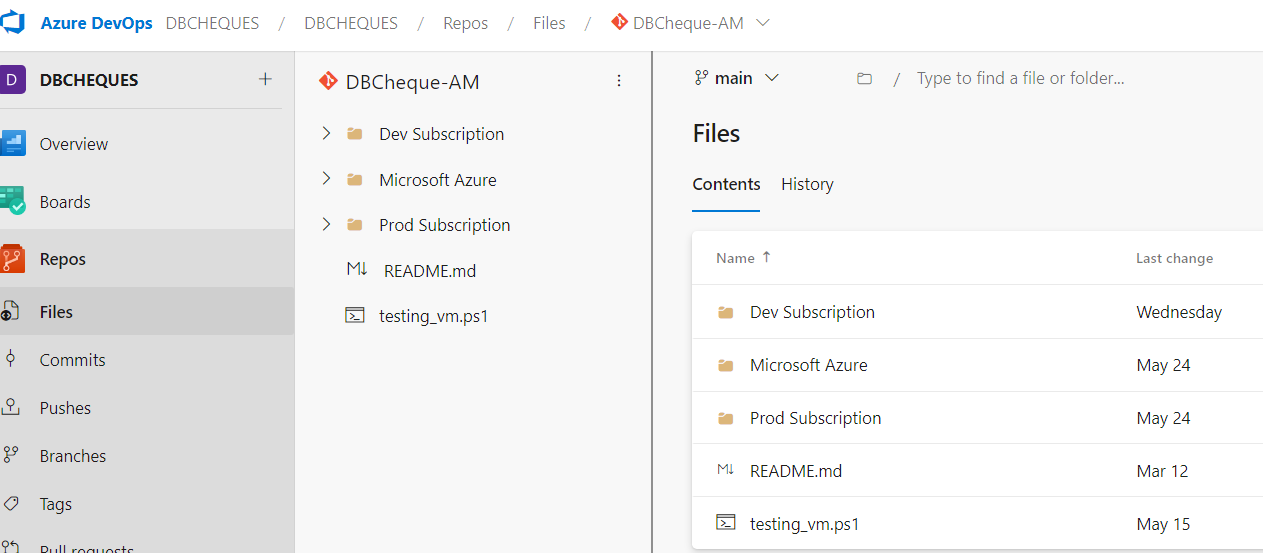




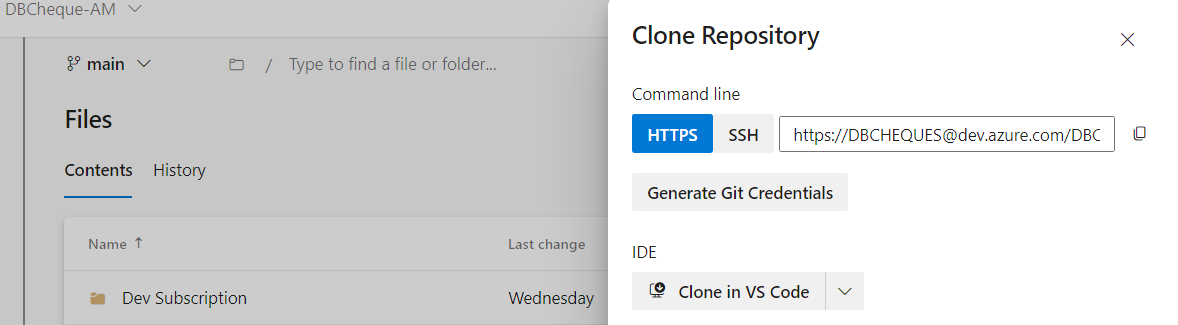
## Create Azure Repos

It is a safe and organized place to store all the code.

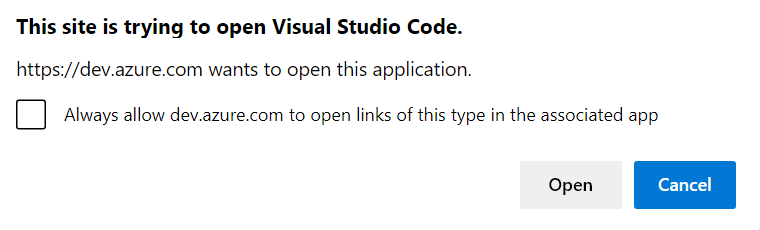
* Go to Repos and then click on Files.



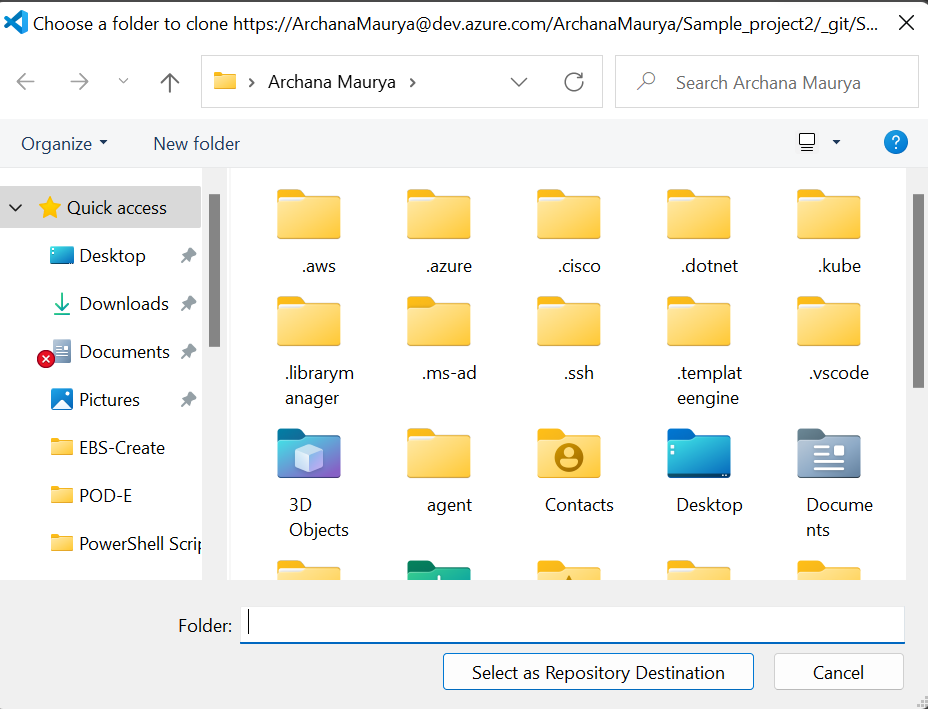
* Steps to clone the new repository created in VS Code
  1. Click on the three dots next to HCL-
  2. After the Clone Repository page will pop-up. As shown below in the screen shot
  3. Click on “**Clone in VS Code”.**



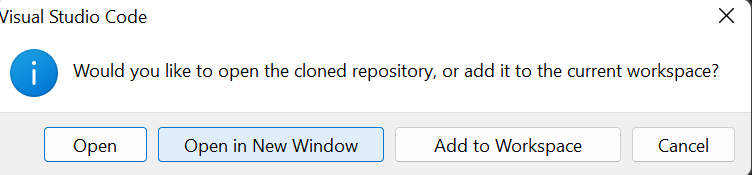
After clicking that you will get a pop-up as shown below



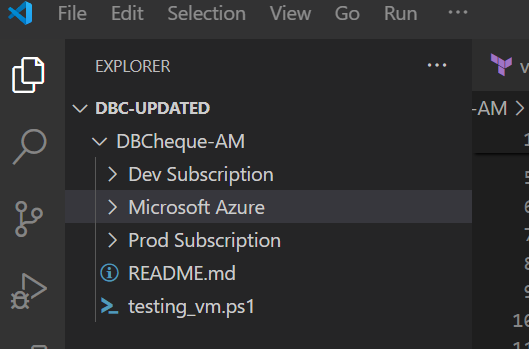
* Select the location in your machine where you want to clone your repository. Click on “**Select as Repository Destination”.**



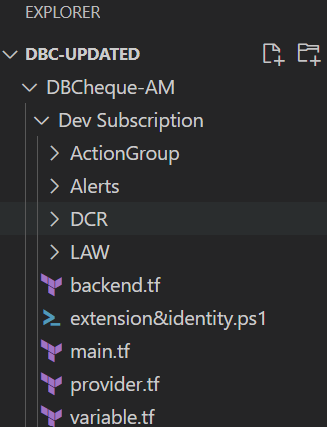
* After the cloning is complete, you’ll be prompted to open the repository. Click “Open in New Window” or “Open” to open the repository in VS Code.



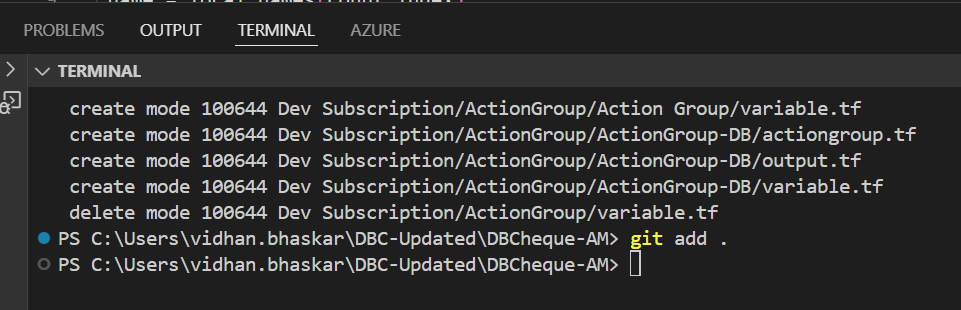
* Now, we will have a local copy of the repository in the specified directory. As shown below



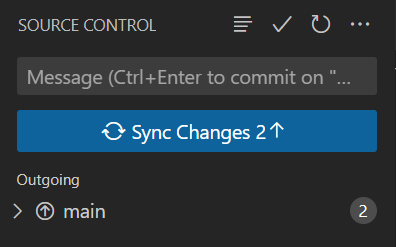
* You can now create your terraform configuration files within the cloned repository by clicking on “New File”.



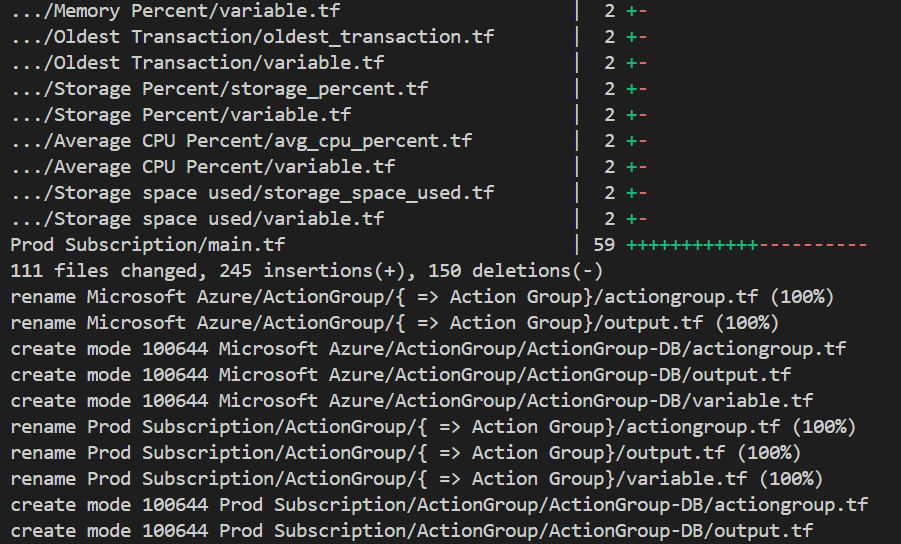
* When you make changes, stage the changes by clicking the “+” button next to each file or using “**git add .**” in the terminal.



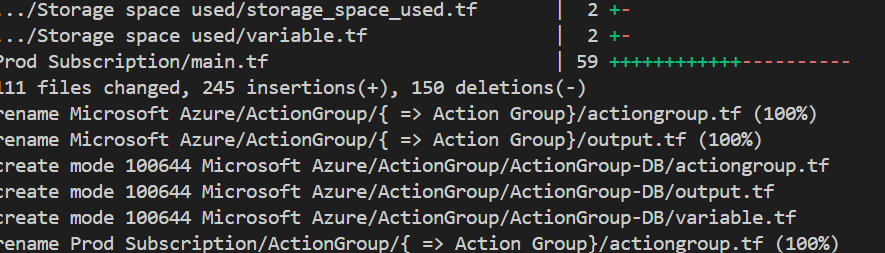
**By Using Git Command (git add.)**

****

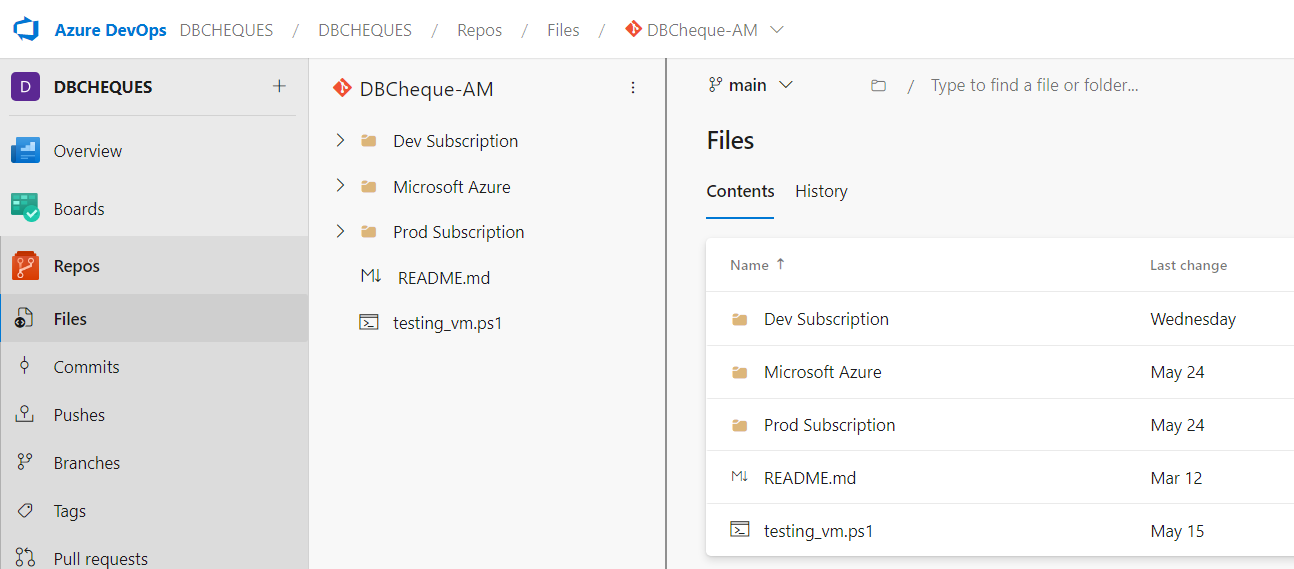
* Enter a commit message and click the checkmark icon or use “**git commit –m ‘Your message’**” in the terminal.



* Use the “**git push”** to push the code into azure repos.

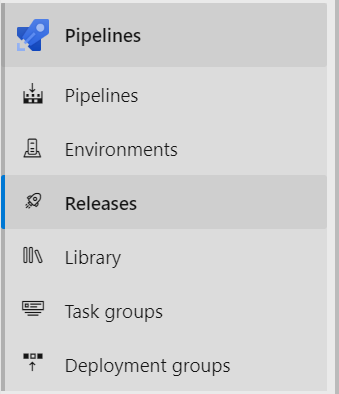


* As you can see on below image, all the changes are reflected on Azure Repos.

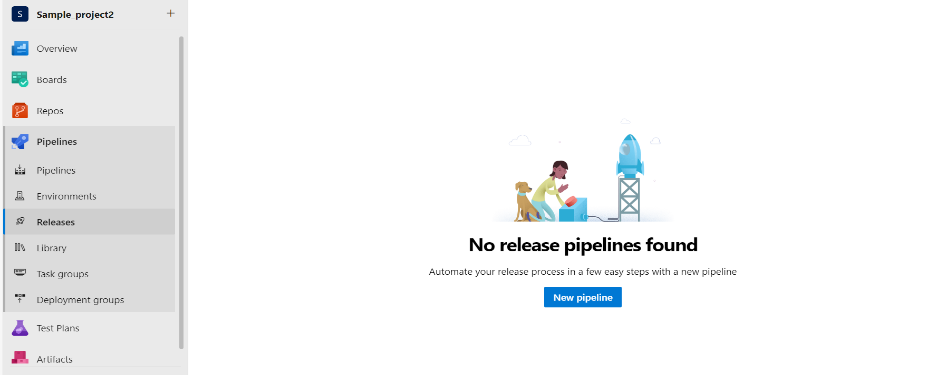
*  To update local repository with changes from Azure Repos, use the “**Pull”** button in the Source Control view or run “**git pull**” in the terminal.

## Create Release Pipelines

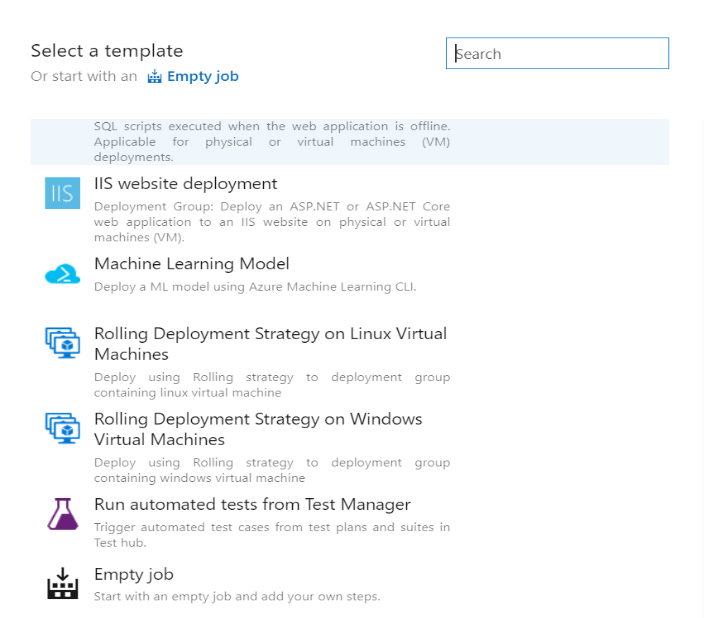
* Go to Pipelines and select “**Releases”.**



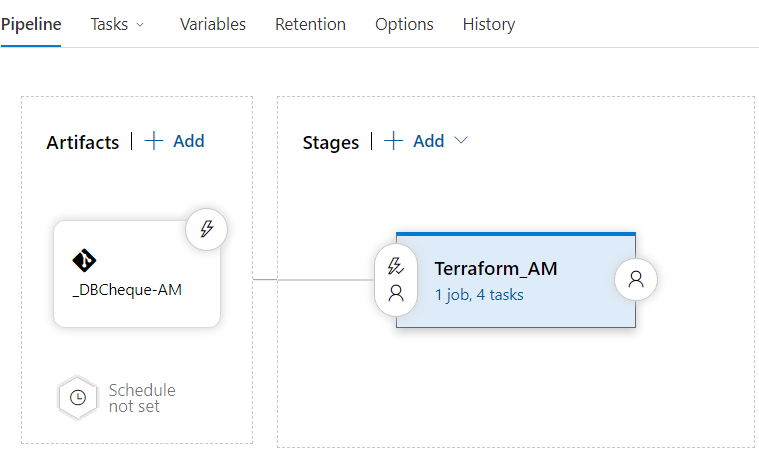
* Click on “**New Pipeline”**.



* Choose the “**Empty Job**” template to start with a blank pipeline configuration.

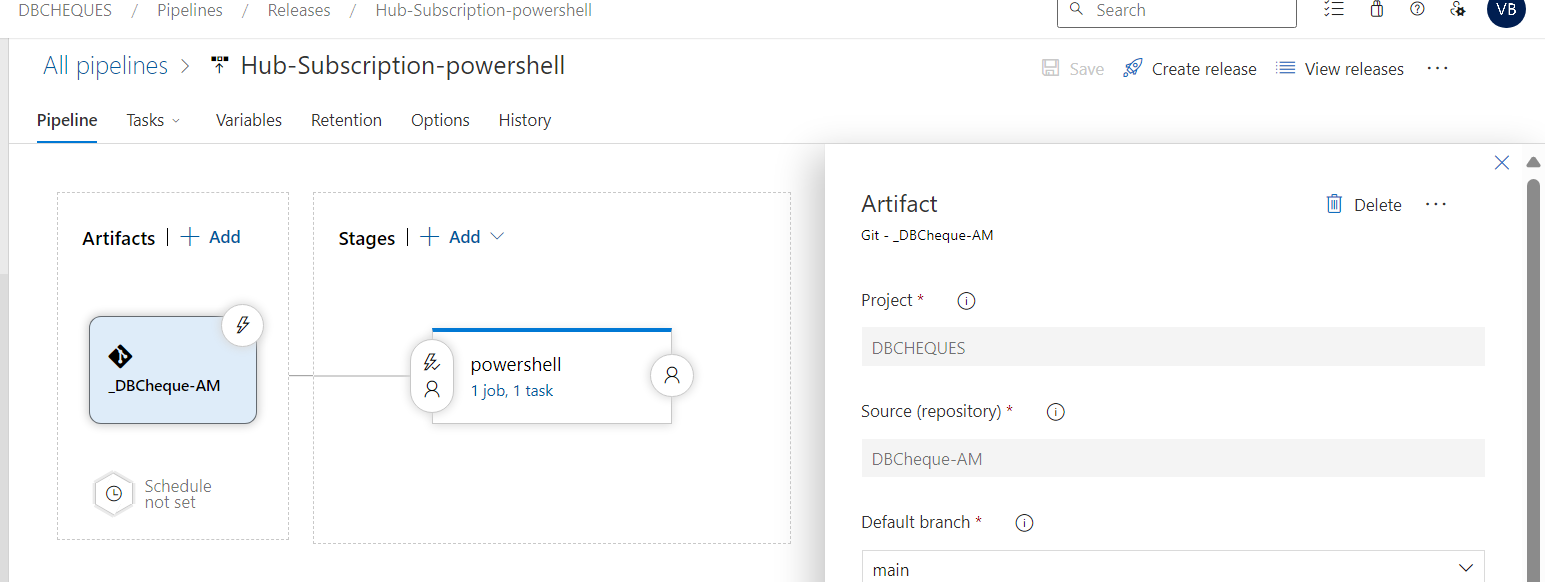


* Create a “**stage**”.

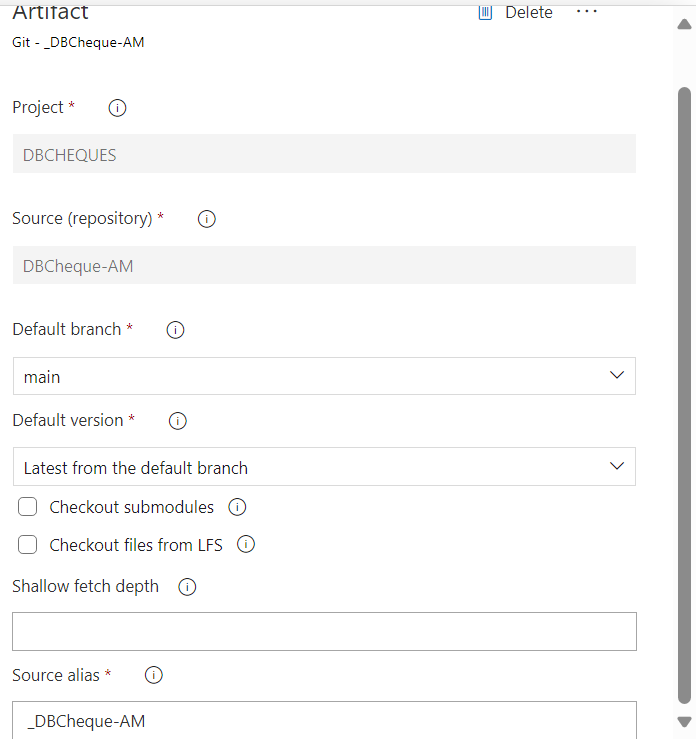


* After creating stage, **add an artifact.**

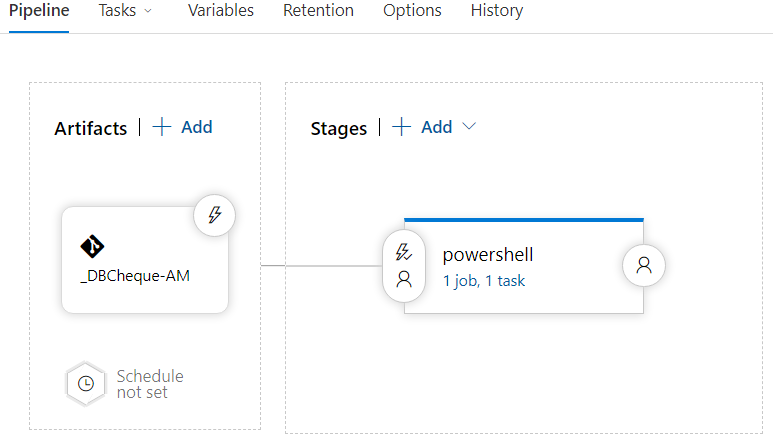




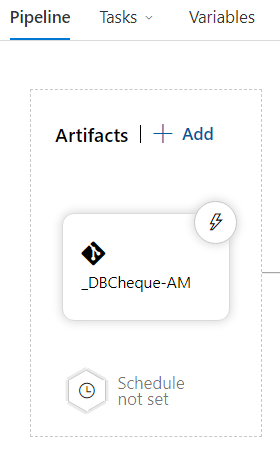
* Click on “Add” button after selecting project, source etc.

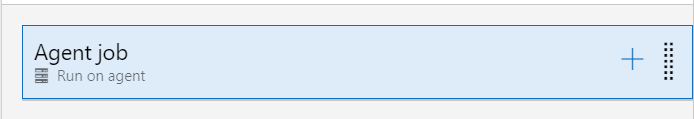


* Here, both Artifact and Stages are added.

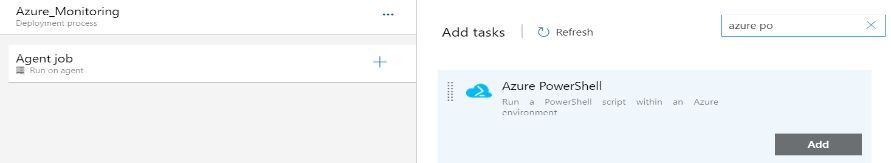


* Go to Agent job.

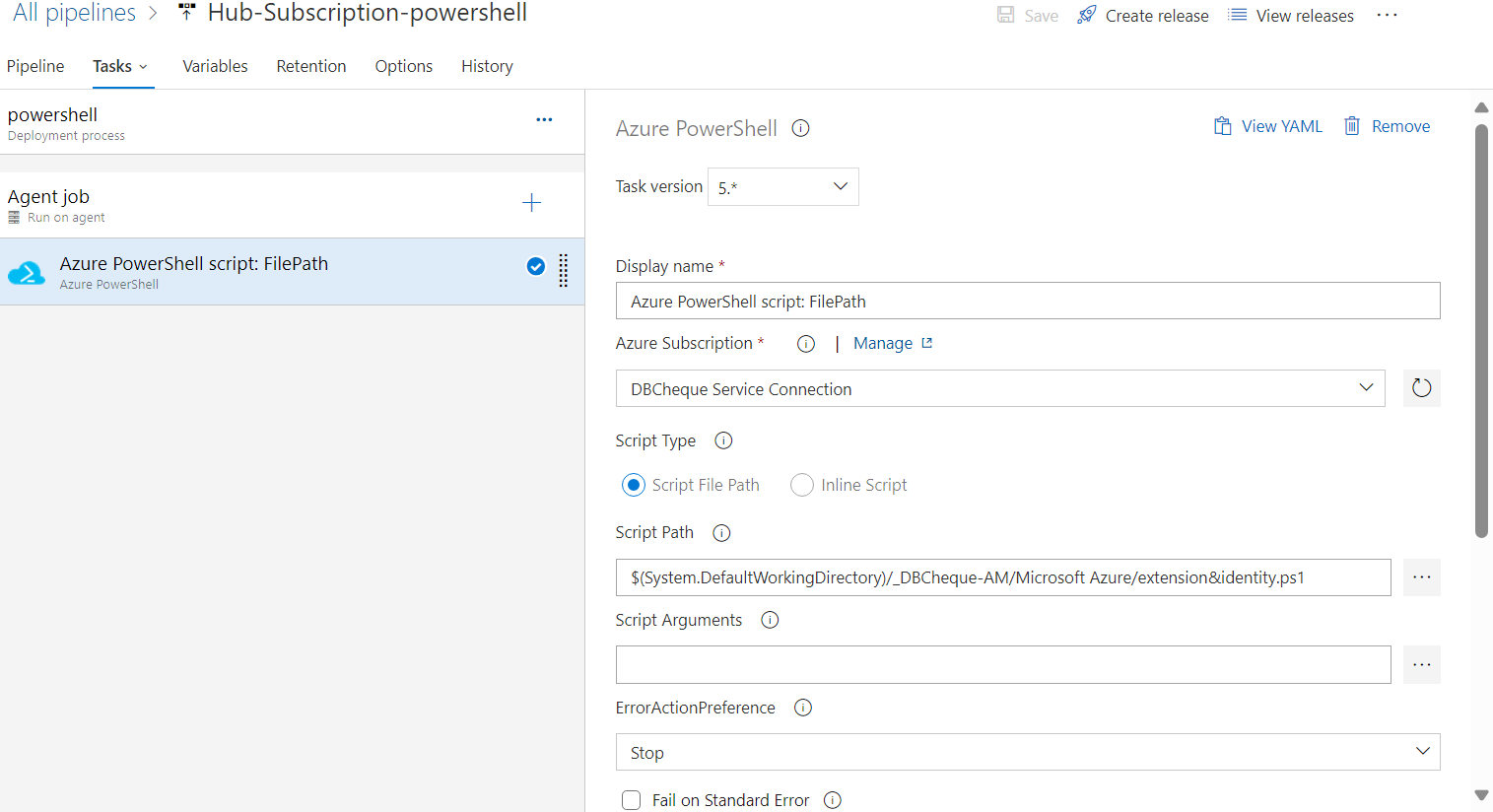




* Add a task into stages.



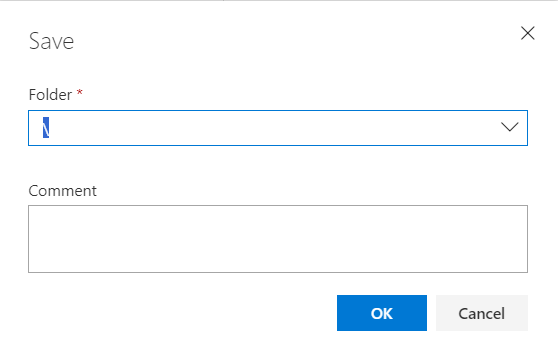
* Select the agent job, Add display name, Azure Subscription, and path of the script.



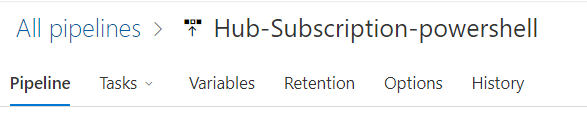
* Save the changes by clicking on “Save” button.



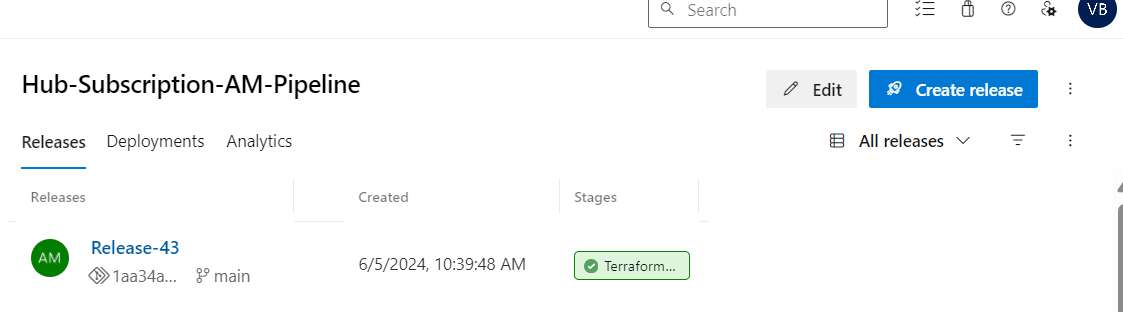
1. Click on “OK”.



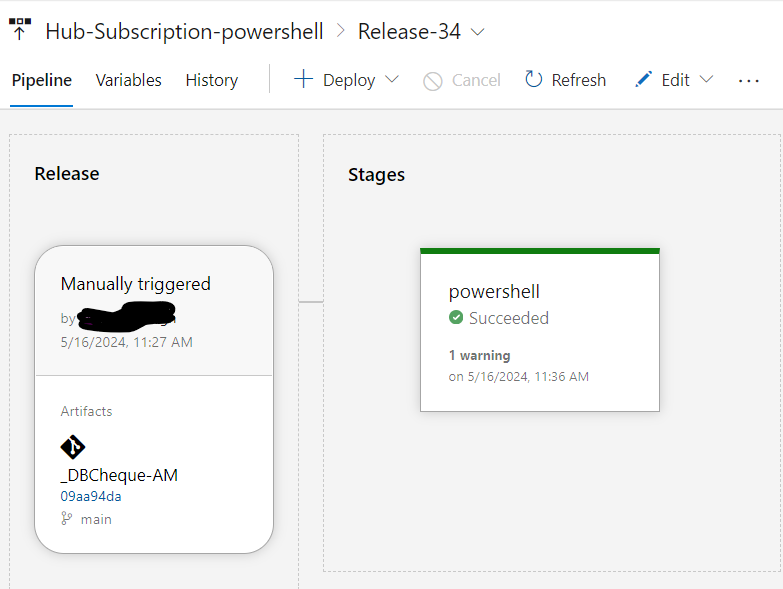
* Click on “**Create release**”.



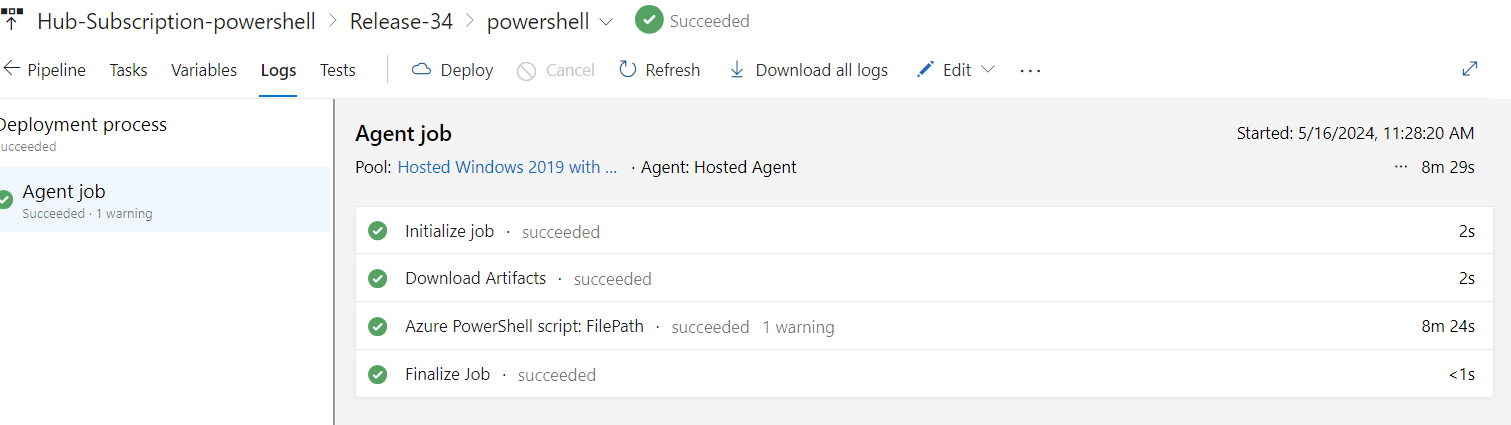
* Click on “Create”.
* Here, our release is created.



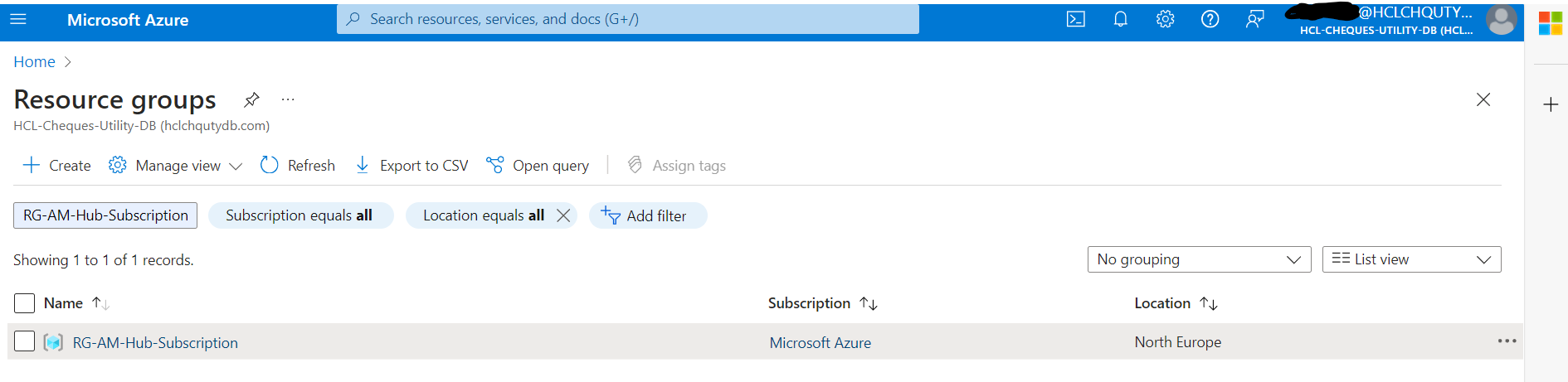
* After creating releases, we need to deploy our script. Click on “Deploy” button.

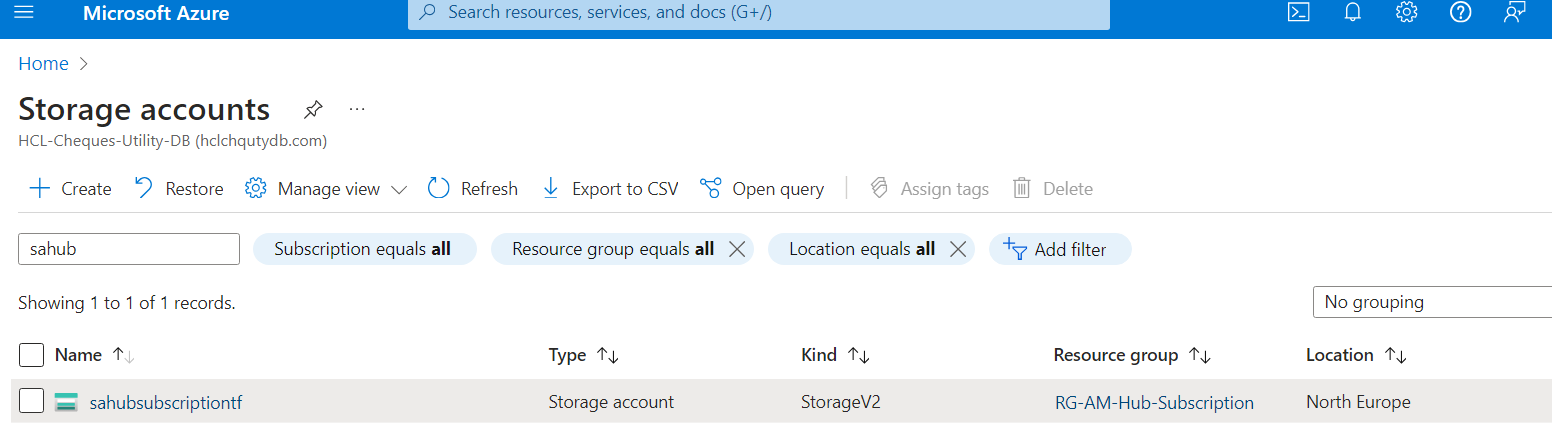


s

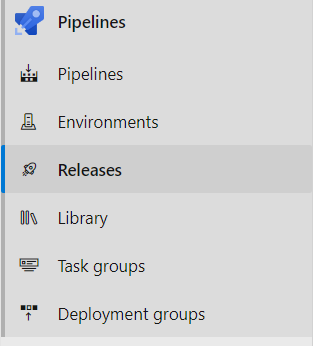


* Here, our resources (resource group, storage account) are created successfully on Azure Portal.

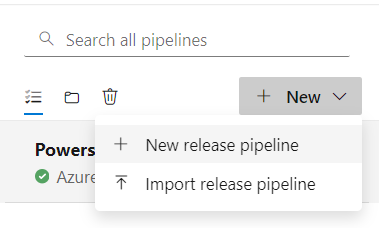




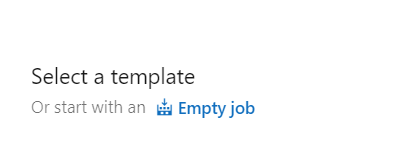
* After creation of resource group and storage account we need to create another pipeline for creating our alerts using terraform.
* Once again, go to pipelines and select Releases.



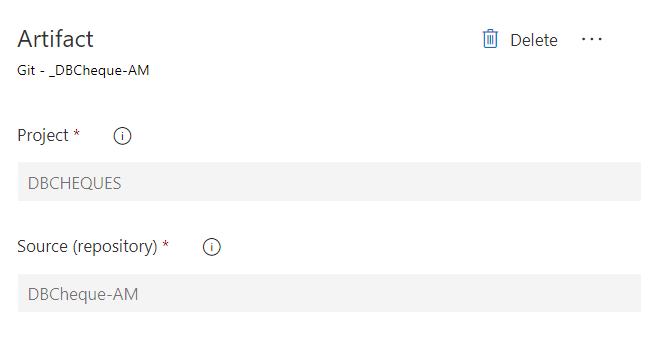
* Click on “New release pipeline” for creation of new release pipeline.



* Choose the “**Empty Job**” template to start with a blank pipeline configuration.



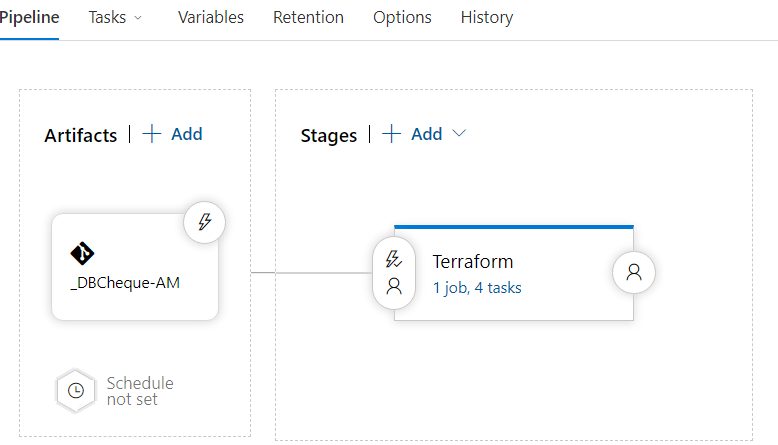
* Create a “**stage**”.



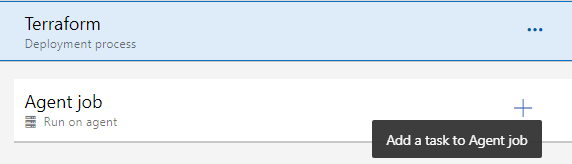
* After creating stage, **add an artifact.**



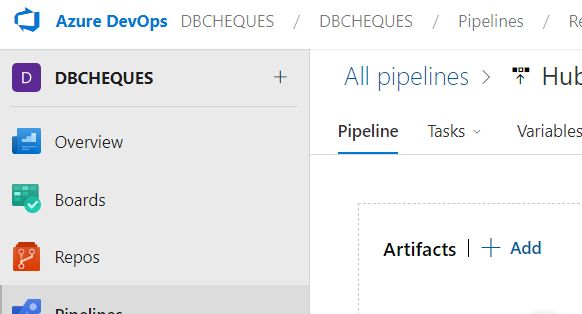
* Here, both Artifact and Stages are added.



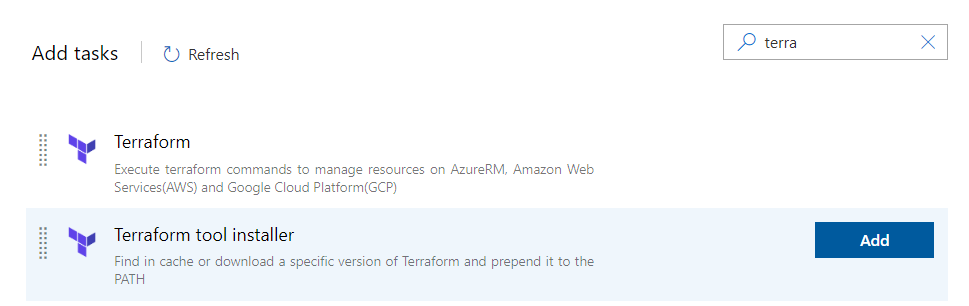
* Go to Agent job.

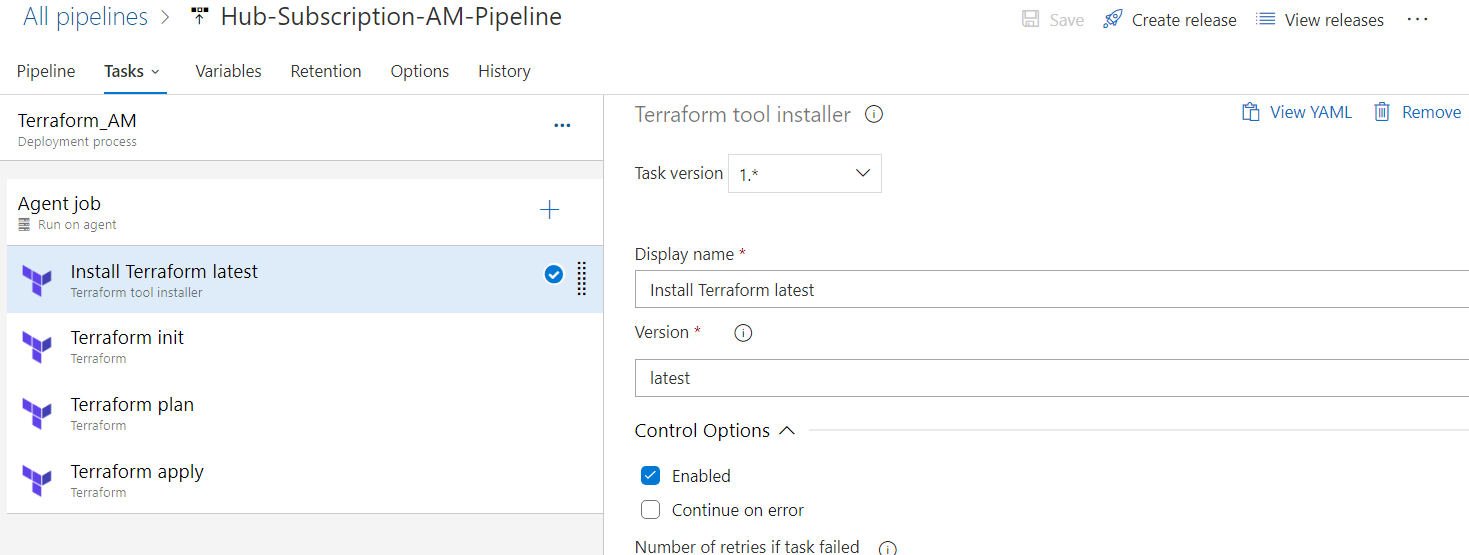


* Click on “+” symbol, for adding different task to Agent job. Search for terraform.
* Select your organization.

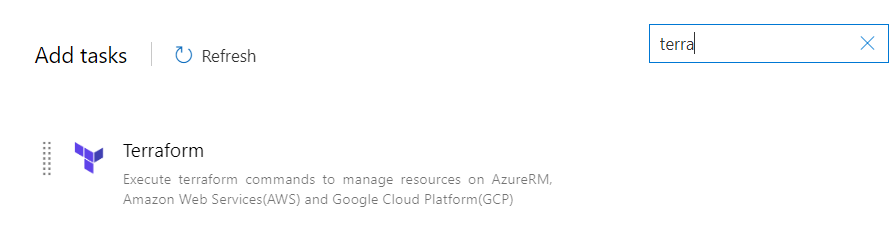


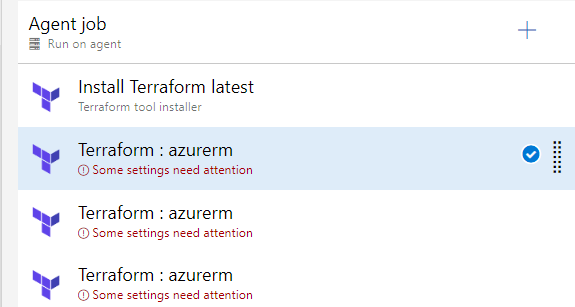
* The first task added is “**Terraform tool installe**r”.



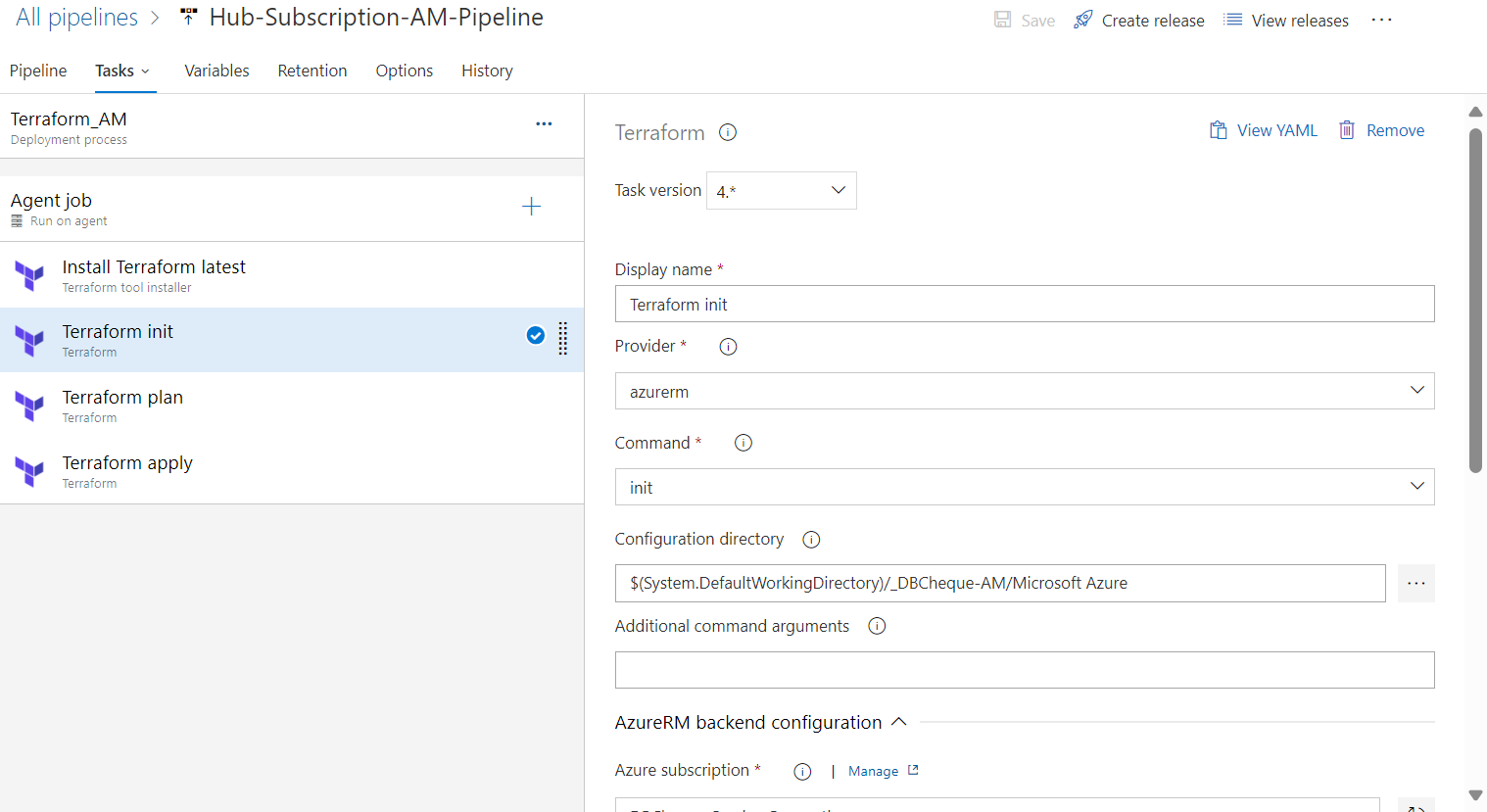


* After adding Terraform tool installer, three instances of terraform were added.
* Again, add another task in Agent Job. Click on **“+**” symbol.
* Search for terraform.



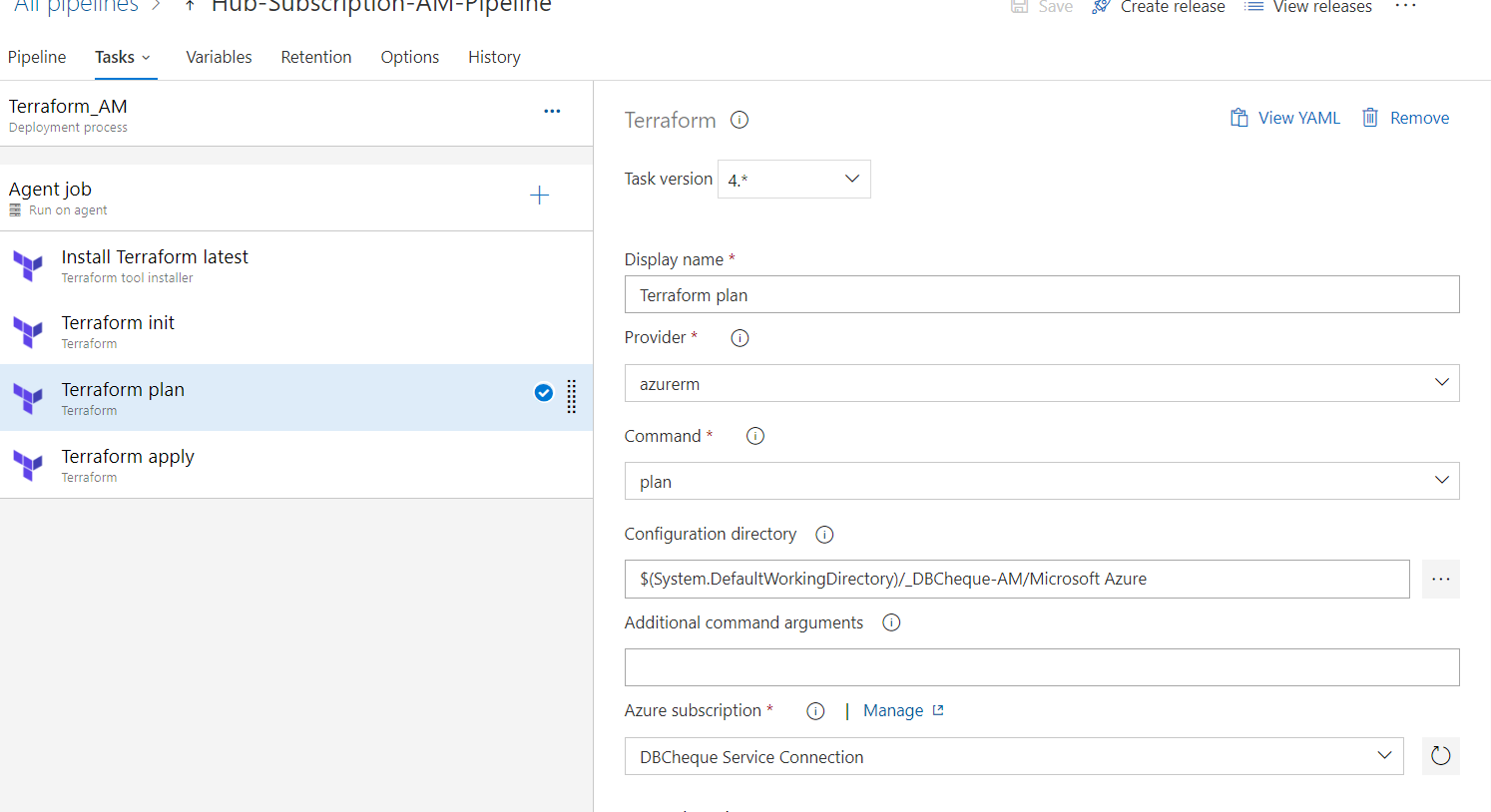


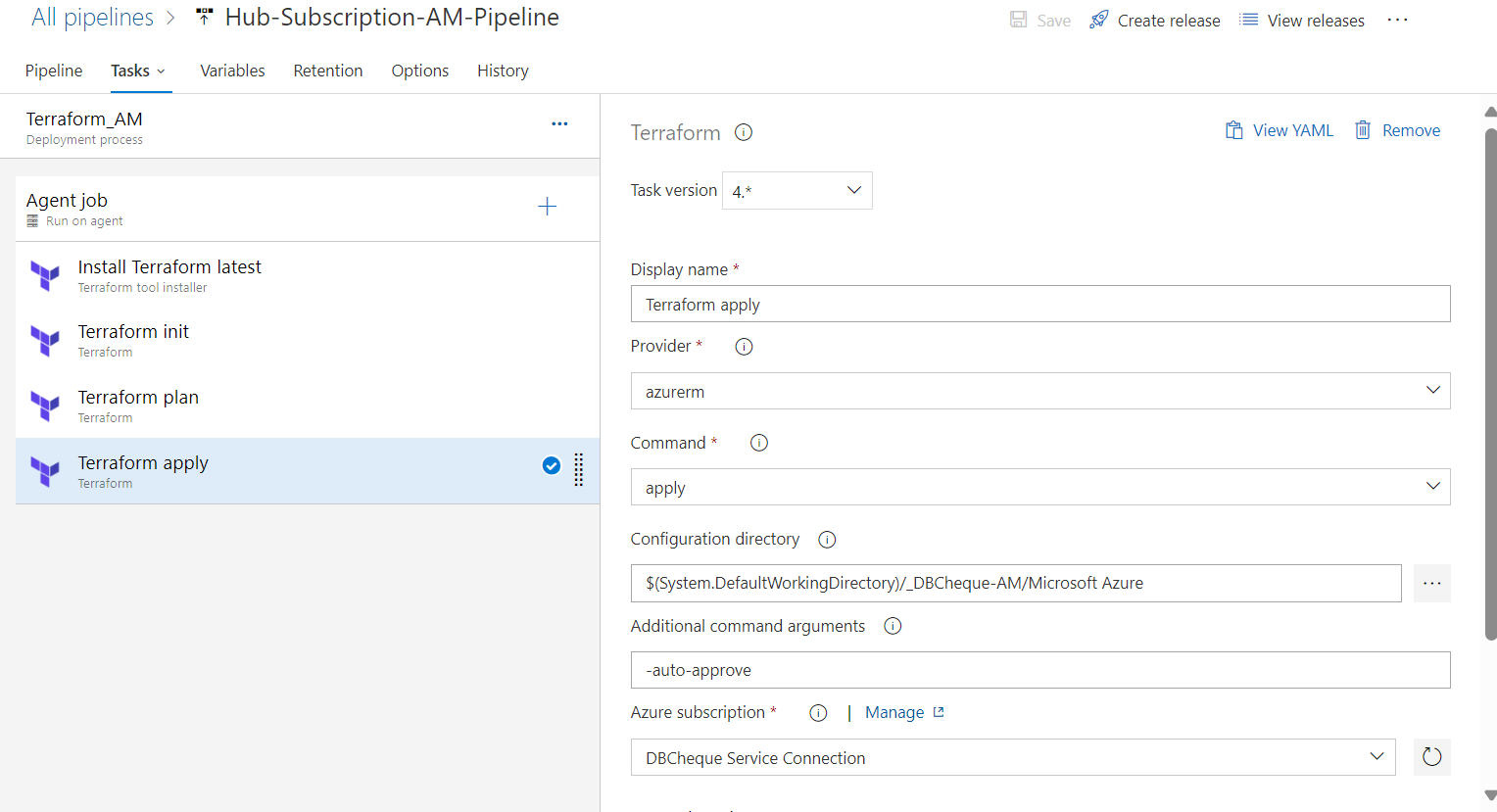
**For terraform init:**



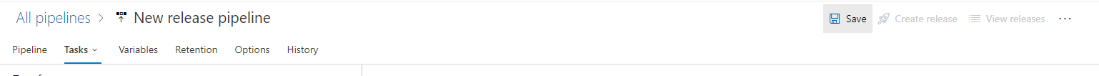


**For terraform plan:**

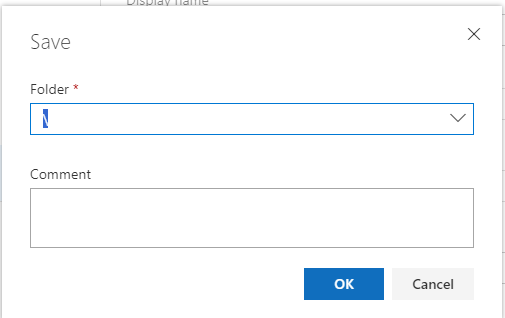




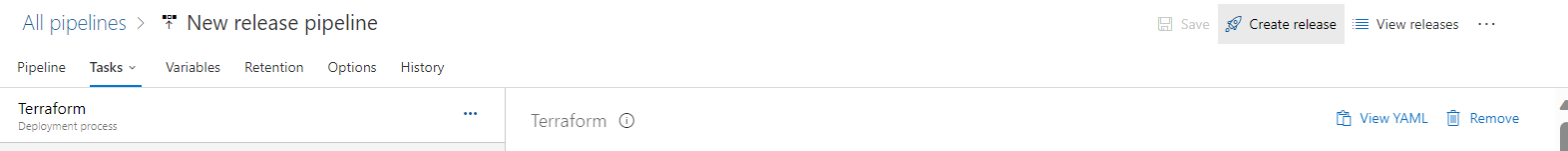
* Save the changes by clicking on “Save” button.

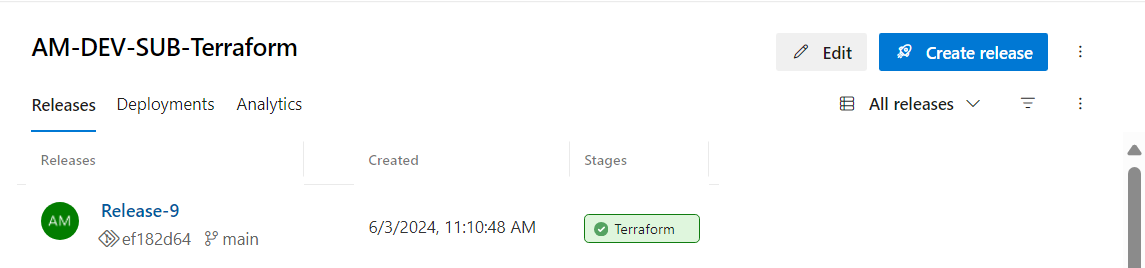


* Click on “OK”.

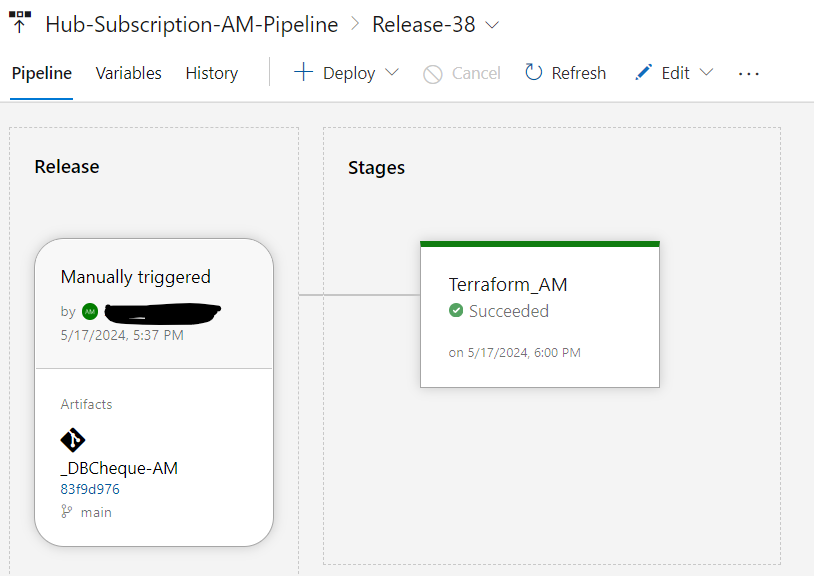


* Click on “**Create release**”.





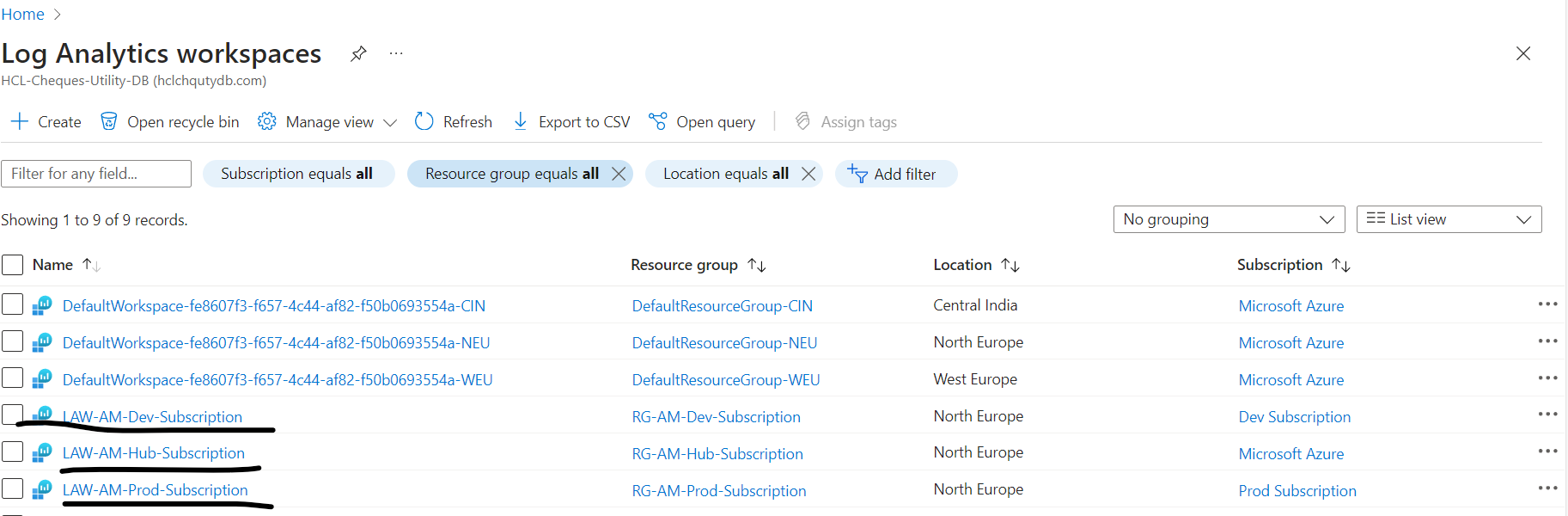
* Click on “Create”.
* Now, our release is created.
* After creating releases, we need to deploy our script. Click on “Deploy” button.



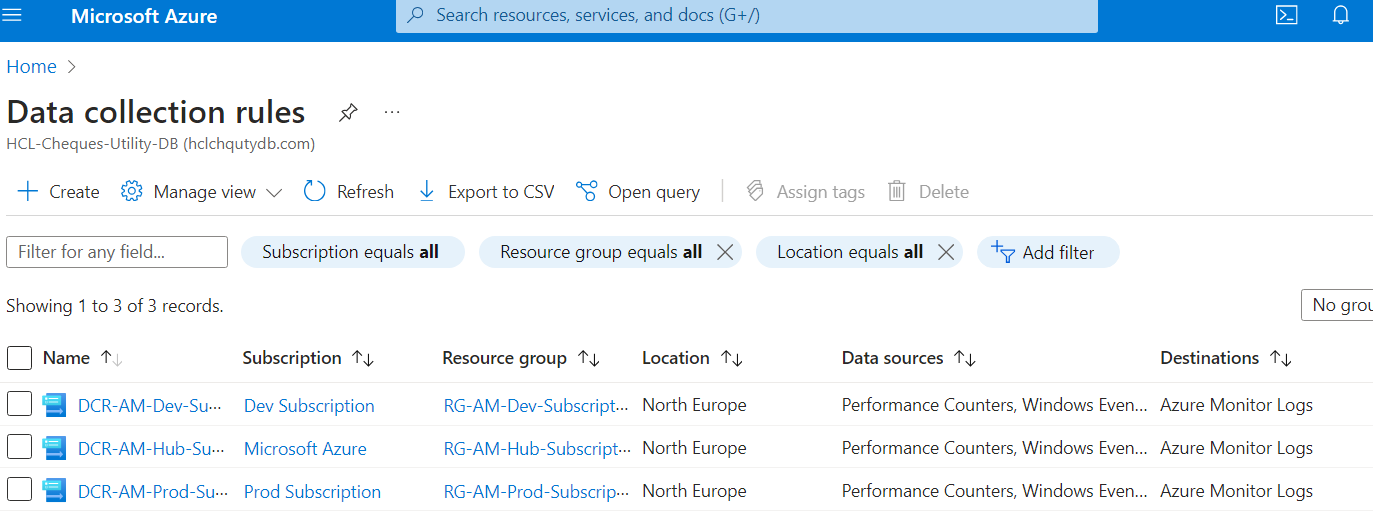
* Here, resources are created on azure portal.

**Outputs:**

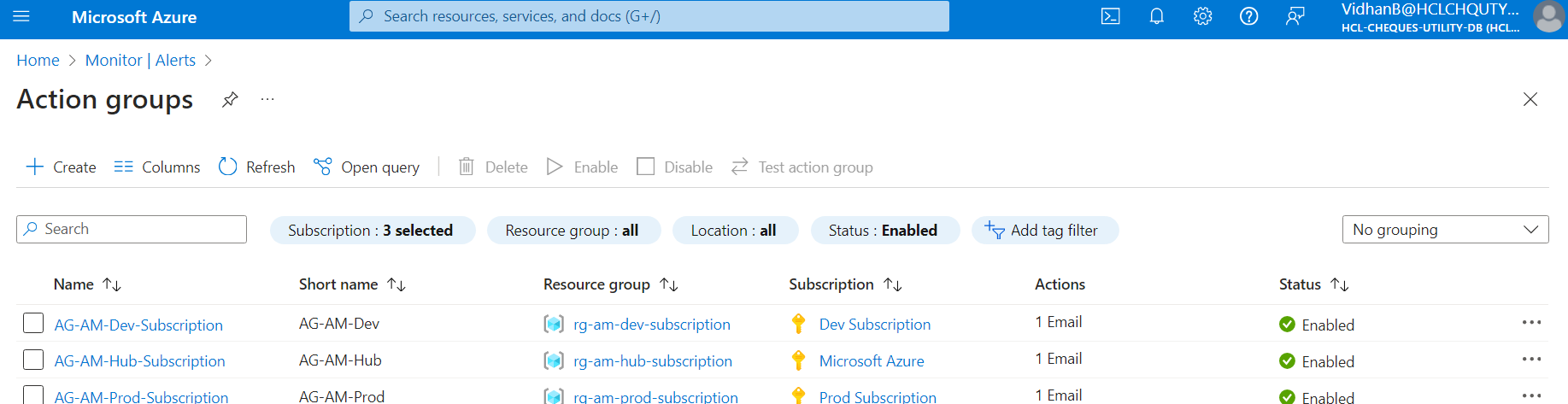
**Log Analytics Workspace (LAW):**

****

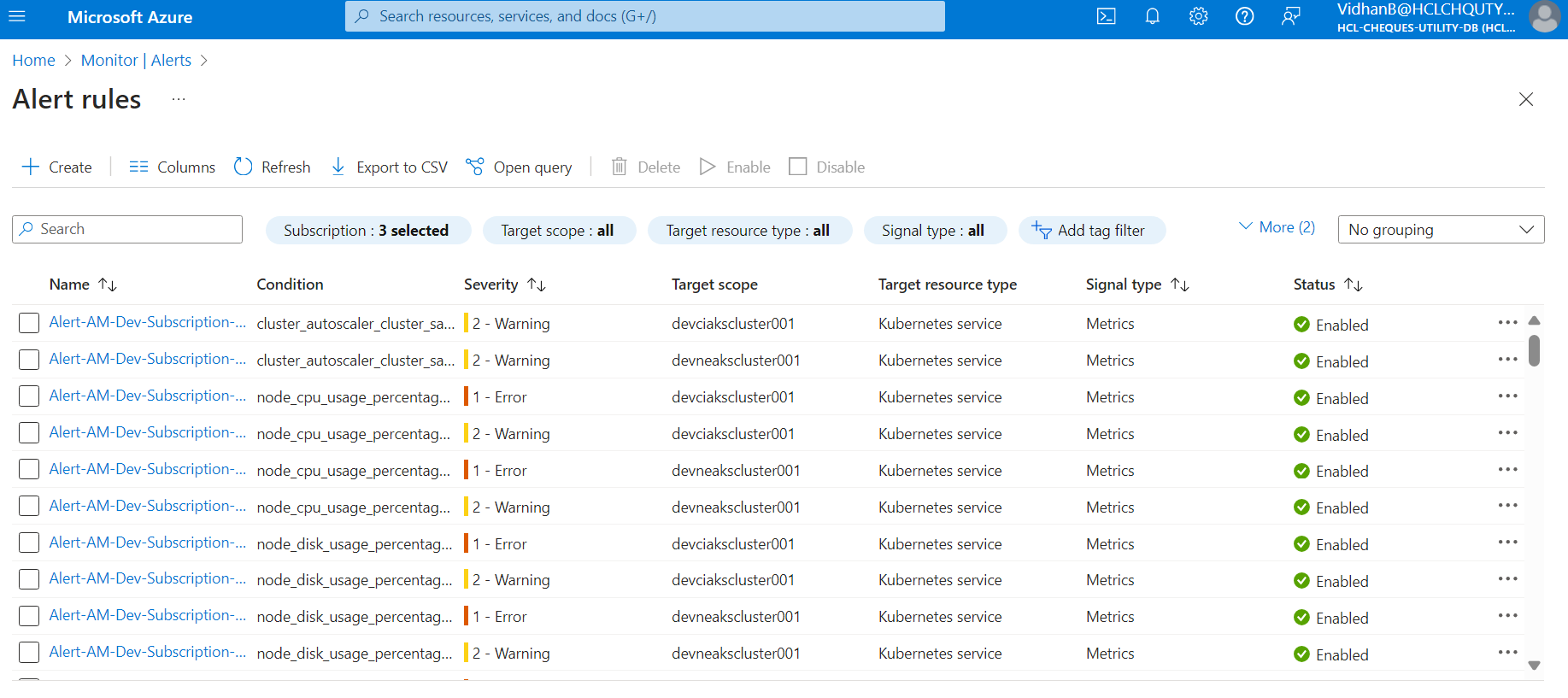
**Data Collection Rule (DCR):**

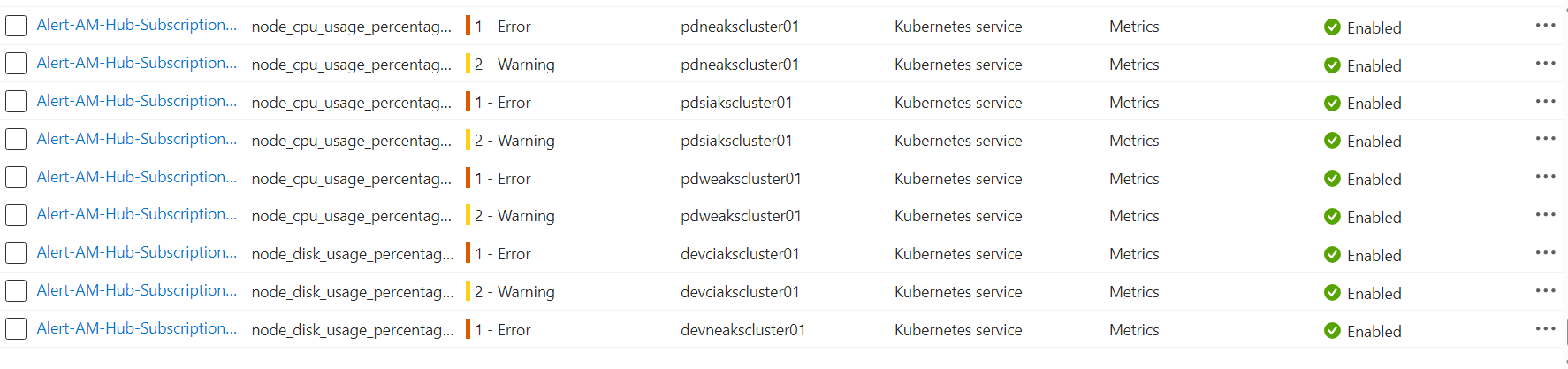


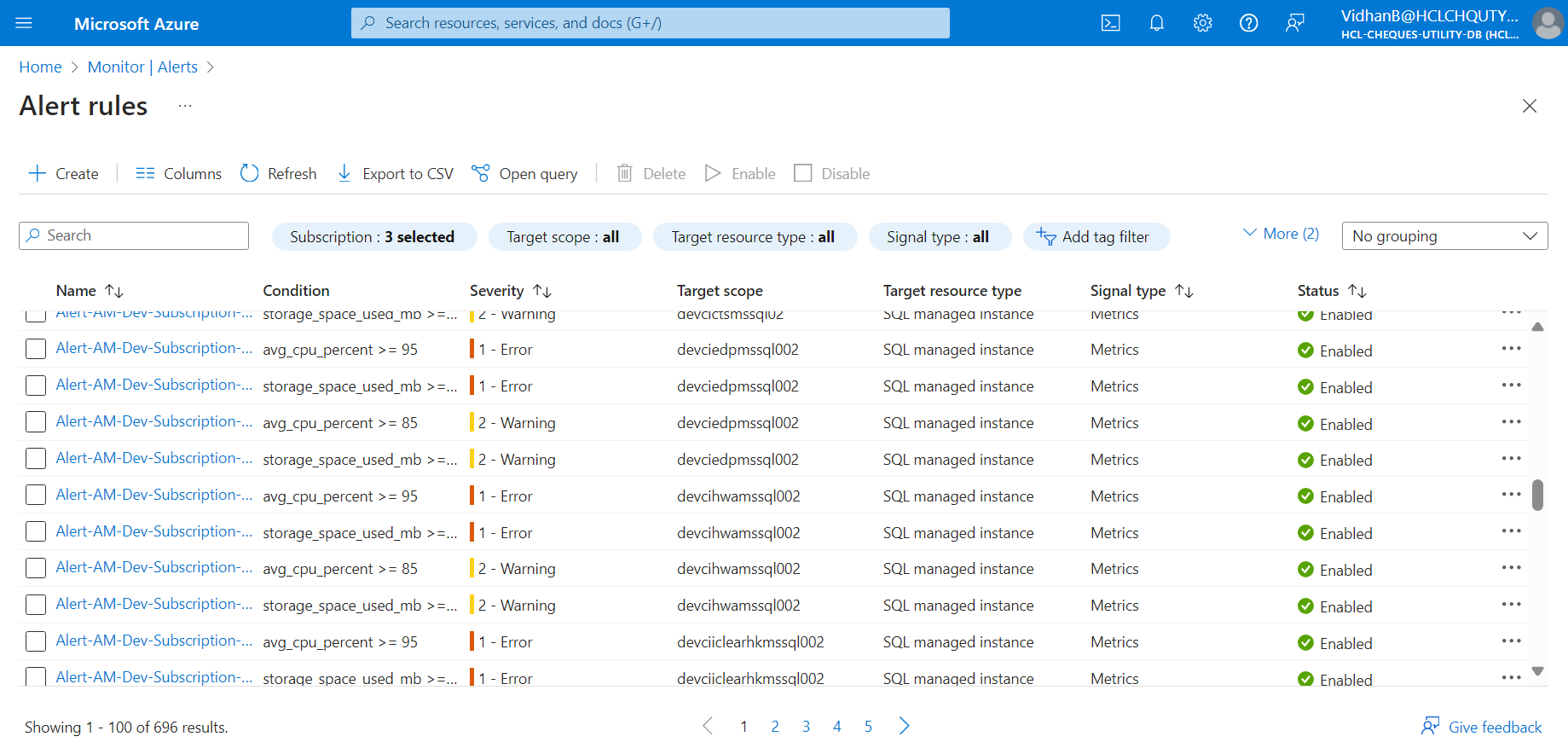
**Action Group:**



**Alerts:**



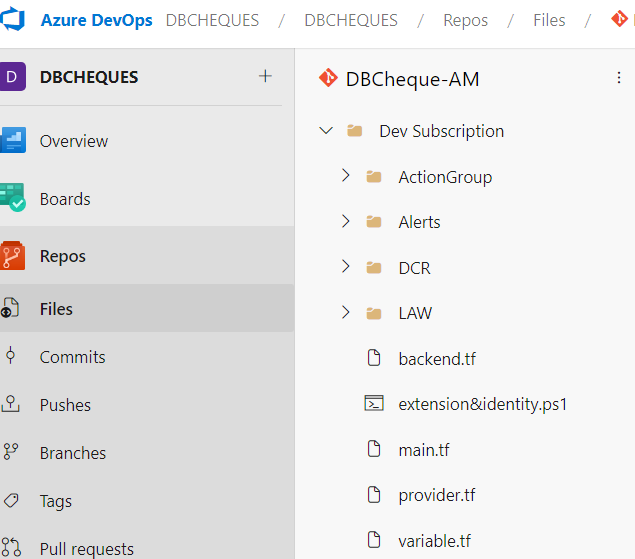


****

## Codes in Azure DevOps Organization

**Step 1**: Create a folder with subscription name. Under that create files as mentioned

* Extension &identity.ps1 - This contains PowerShell script which will create a resource group, also will enable the system identity, and will install extension on the basis of operating system for virtual machines.
* variable.tf - Storing resource group and virtual machines details.
* backend.tf - This has the details where our backend state file will be stored
* Provider.tf - This contains subscription details
* main.tf - This helps in calling modules.



* Below is the PowerShell code for extension & identity.ps1

|  |
| --- |
| $sub = Get-AzSubscription -SubscriptionName 'Dev Subscription'  Set-AzContext -Subscription $sub.Id  $RGName = "RG-AM-Dev-Subscription"  $storageAccountName = "sadevsubscriptiontf"  $location = "North Europe"  # New-AzResourceGroup -Name $RGName -Location $location  # New-AzStorageAccount -Name $storageAccountName -ResourceGroupName $RGName -SkuName "Standard\_LRS" -Location $location  $ip\_address = @("Virtual machine")  $vms= @()  foreach ($ip in $ip\_address) {      $nic = Get-AzNetworkInterface | Where-Object {$\_.IpConfigurations.PrivateIpAddress -eq $ip}      $vmId = $nic.VirtualMachine.Id      $vms += (Get-AzVM | Where-Object {$\_.Id -eq $vmId})    }      foreach($vm in $vms){          if($vm.Identity.Type){              write-output "$($vm.Identity.Type) is already enabled on $($vm.Name)"               }             else{               write-output "Enabling system assigned Identity on $($vm.Name)"               Update-AzVM -VM $vm -IdentityType SystemAssigned -ResourceGroupName $vm.ResourceGroupName               }      #write-output $vm.StorageProfile.OsDisk.OsType      $statuscheck = Get-AzVM -ResourceGroupName $vm.ResourceGroupName -Name $vm.Name -Status      if ($statuscheck.Statuses.DisplayStatus[1] -eq "VM running") {          $extension = Get-AzVMExtension -VMName $vm.Name -ResourceGroupName $vm.ResourceGroupName          if ($extension.ExtensionType -eq "AzureMonitorLinuxAgent" -or $extension.ExtensionType -eq "AzureMonitorWindowsAgent") {              $extensionName = $extension.ExtensionType              Write-Output "$extensionName already exists."          }          else {              if ($vm.StorageProfile.OsDisk.OsType -eq "Windows") {                  write-output "windows"                  write-output $vm.Name                  write-output $vm.Location                  write-output $vm.ResourceGroupName                  Set-AzVMExtension -Name AzureMonitorWindowsAgent -ExtensionType AzureMonitorWindowsAgent -Publisher Microsoft.Azure.Monitor -ResourceGroupName $vm.ResourceGroupName -VMName $vm.Name -Location $vm.Location -TypeHandlerVersion 1.6              }              else {                  write-output "linux"                  write-output $vm.Name                  write-output $vm.Location                  write-output $vm.ResourceGroupName                  Set-AzVMExtension -Name AzureMonitorLinuxAgent -ExtensionType AzureMonitorLinuxAgent -Publisher Microsoft.Azure.Monitor -ResourceGroupName $vm.ResourceGroupName -VMName $vm.Name -Location $vm.Location -TypeHandlerVersion 1.6              }          }      } |

**Provider.tf:** A provider is a plugin that Terraform uses to create and manage your resources. We are specifying provider as a **“azurerm”** for creating azure resources and we are taking subscription id of Microsoft Azure account.

Here is the terraform script:

|  |
| --- |
| provider "azurerm" {  features {}  subscription\_id = "\*\*\*\*\*-\*\*\*-\*\*\*-\*\*\*-\*\*\*\*\*\*\*\*\*\*\*\*"  skip\_provider\_registration = true  } |

**variable.tf:** These are used to define and store values that can be reused across our configuration.

Here is the terraform script:

|  |
| --- |
| variable "rg\_name" {    default = "RG-AM-Dev-Subscription"  }  variable "virtual-machines" {    type = map(any)    default = {      vms = [{        vm\_name = "devciaksjump002",        resource\_group\_name ="DEV-CI-AKS-RG-GB-001",        os\_type = "linux"      },      {        vm\_name = "DEVCICTS00001",        resource\_group\_name ="dev-ci-cts-rg-001",        os\_type = "windows"      },      {        vm\_name = "DEVCICTS02",        resource\_group\_name ="dev-ci-cts-rg-001",        os\_type = "windows"      },      {        vm\_name = "devcidboracle1",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "linux"      },      {        vm\_name = "devcidboracle2",        resource\_group\_name ="dev-ci-gis-rg-002",        os\_type = "linux"      },      {        vm\_name = "DEVCIDEVIAC002",        resource\_group\_name ="dev-ci-iac-rg-001",        os\_type = "linux"      },      {        vm\_name = "devcielk00001",        resource\_group\_name ="dev-ci-apacil-rg-001",        os\_type = "linux"      },      {        vm\_name = "devciexacto0001",        resource\_group\_name ="dev-ci-iclear-rg-001",        os\_type = "linux"      },      {        vm\_name = "devciexacto0002",        resource\_group\_name ="dev-ci-iclear-rg-001",        os\_type = "linux"      },      {        vm\_name = "devcigis00001",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {        vm\_name = "devcihwa00001",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "linux"      },      {        vm\_name = "DEVCIICLCTWIN01",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "windows"      },      {        vm\_name = "devcijasper0001",        resource\_group\_name ="dev-ci-iclear-rg-001",        os\_type = "linux"      },      {        vm\_name = "devcijenkins01",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "linux"      },      {        vm\_name = "DEVCISFTP00001",        resource\_group\_name ="dev-ci-apacil-rg-001",        os\_type = "windows"      },      {        vm\_name = "devcisftp11",        resource\_group\_name ="dev-ci-apacil-rg-001",        os\_type = "linux"      },      {        vm\_name = "devneaksjump002",        resource\_group\_name ="dev-ne-aks-rg-gb-001",        os\_type = "linux"      },      {        vm\_name = "devnechk00001",        resource\_group\_name ="dev-ne-chk-rg-001",        os\_type = "linux"      },      {        vm\_name = "devneelk00001",        resource\_group\_name ="dev-ne-usil-rg-001",        os\_type = "linux"      },      {        vm\_name = "DEVNEIDM00001",        resource\_group\_name ="dev-ne-idmcre-rg-001",        os\_type = "windows"      },      {        vm\_name = "DEVNESFTP00001",        resource\_group\_name ="dev-ne-apacil-rg-001",        os\_type = "windows"      },      {        vm\_name = "devnesftp11",        resource\_group\_name ="dev-ne-apacil-rg-001",        os\_type = "linux"      },      {        vm\_name = "Precigis00001",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {        vm\_name = "preciiclct001",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },      {        vm\_name = "preciiclct002",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },      {        vm\_name = "PRFTSTCIAPSCN1",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },      {        vm\_name = "PRFTSTCIAPSCN2",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },      {        vm\_name = "rubrikpocaks-vm-0",        resource\_group\_name ="dev-ci-rubrik-rg-001",        os\_type = "linux"      },      {        vm\_name = "SITCIICHQ00001",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },      {        vm\_name = "sitciiclct001",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },      {        vm\_name = "sitciiclct002",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },      {        vm\_name = "SITNEICHQ00001",        resource\_group\_name ="dev-ne-hwa-rg-001",        os\_type = "windows"      },      {        vm\_name = "UATCIGISAUTO004",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {        vm\_name = "UATCIGISAUTO005",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {        vm\_name = "UATCIGISAUTO006",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {        vm\_name = "uatciiclct001",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },      {        vm\_name = "UATNEIDM00001",        resource\_group\_name ="dev-ne-idmcre-rg-001",        os\_type = "windows"      }      ]    }  } |

**backend.tf:** We use this to configure the backend infrastructure where terraform should store its state files.

Here is the terraform script:

|  |
| --- |
| terraform {    backend "azurerm" {        resource\_group\_name  = "RG-AM-Dev-Subscription"        storage\_account\_name = "sadevsubscriptiontf"        container\_name       = "new-container"        key                  = "terraform.tfstate"    }  } |

Here is the terraform code for main.tf

|  |
| --- |
| module "law" {    source  = "./LAW"    rg\_name = var.rg\_name  }  module "ActionGroup" {    source = "./ActionGroup/Action Group"    rg\_name = var.rg\_name  }  module "ActionGroup-DB" {    source = "./ActionGroup/ActionGroup-DB"    rg\_name = var.rg\_name  }  module "SQL-ManagedInstance-CPUPercent" {    source = "./Alerts/SQL Managed Instances/Average CPU Percent"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "SQL-ManagedInstance-StorageSpaceUsed" {    source = "./Alerts/SQL Managed Instances/Storage space used"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-Availability-CI" {    source = "./Alerts/PostgreSQL-FlexServer-CentralIndia/Availability"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-CPUPercent-CI" {    source = "./Alerts/PostgreSQL-FlexServer-CentralIndia/CPU Percent"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-MaximumConnection-CI" {    source = "./Alerts/PostgreSQL-FlexServer-CentralIndia/Max Connections"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-MemoryPercent-CI" {    source = "./Alerts/PostgreSQL-FlexServer-CentralIndia/Memory Percent"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-OldestTransaction-CI" {    source = "./Alerts/PostgreSQL-FlexServer-CentralIndia/Oldest Transaction"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-StoragePercent-CI" {    source = "./Alerts/PostgreSQL-FlexServer-CentralIndia/Storage Percent"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-Availability-NE" {    source = "./Alerts/PostgreSQL-Flex Server-North Europe/Availability"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-CPUPercent-NE" {    source = "./Alerts/PostgreSQL-Flex Server-North Europe/CPU Percent"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-MaximumConnection-NE" {    source = "./Alerts/PostgreSQL-Flex Server-North Europe/Max Connections"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-MemoryPercent-NE" {    source = "./Alerts/PostgreSQL-Flex Server-North Europe/Memory Percent"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-OldestTransaction-NE" {    source = "./Alerts/PostgreSQL-Flex Server-North Europe/Oldest Transaction"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "PostgreSQL-Flex-StoragePercent-NE" {    source = "./Alerts/PostgreSQL-Flex Server-North Europe/Storage Percent"    action\_group\_db\_id = module.ActionGroup-DB.action\_group\_db\_id    rg\_name = var.rg\_name  }  module "DCR" {    source                = "./DCR"    rg\_name               = var.rg\_name    virtual\_machines      = var.virtual-machines    workspace\_resource\_id = module.law.log\_id  }  module "Windows-CPUUtil" {    source                = "./Alerts/WindowsAlerts/CPUUtilizationAlert"    rg\_name               = var.rg\_name    workspace\_resource\_id = module.law.log\_id    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Windows-DataDiskIOPS-Central" {    source                = "./Alerts/WindowsAlerts/DataDiskIOPS - Central India"    rg\_name               = var.rg\_name    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Windows-DataDiskIOPS-North" {    source                = "./Alerts/WindowsAlerts/DataDiskIOPS - North Europe"    rg\_name               = var.rg\_name    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Windows-LogicalDiskFreeSpace" {    source                = "./Alerts/WindowsAlerts/LogicalDiskFreeSpace"    rg\_name               = var.rg\_name    workspace\_resource\_id = module.law.log\_id    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Windows-MemoryAvailableBytes" {    source                = "./Alerts/WindowsAlerts/MemoryAvailableBytes"    rg\_name               = var.rg\_name    workspace\_resource\_id = module.law.log\_id    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Windows-SystemEventLog" {    source                = "./Alerts/WindowsAlerts/System Event Log ShutdownStart"    rg\_name               = var.rg\_name    workspace\_resource\_id = module.law.log\_id    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "VMAvailability-CentralIndia" {  source="./Alerts/VMAvailabilityAlert - CI"  rg\_name=var.rg\_name  action\_group\_id = module.ActionGroup.action\_group\_id  }  module "VMAvailability-NorthEurope" {  source="./Alerts/VMAvailabilityAlert - NE"  rg\_name=var.rg\_name  action\_group\_id = module.ActionGroup.action\_group\_id  }  module "Linux-CPUUtilization" {    source                = "./Alerts/LinuxAlerts/CPUUtilization"    rg\_name               = var.rg\_name    workspace\_resource\_id = module.law.log\_id    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Linux-LogicalDiskUsedSpace" {    source                = "./Alerts/LinuxAlerts/LogicalDiskUsedSpace"    rg\_name               = var.rg\_name    workspace\_resource\_id = module.law.log\_id    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Linux-MemoryUsedPercentage" {    source                = "./Alerts/LinuxAlerts/Memory Used Percentage"    rg\_name               = var.rg\_name    workspace\_resource\_id = module.law.log\_id    action\_group\_id       =module.ActionGroup.action\_group\_id  }  module "Linux-MemoryUsedSwapSpace" {    source                = "./Alerts/LinuxAlerts/Memory Used Swap Space"    rg\_name               = var.rg\_name    workspace\_resource\_id = module.law.log\_id    action\_group\_id       = module.ActionGroup.action\_group\_id  }    module "Cluster-Health" {    source                = "./Alerts/Azure\_Kubernetes\_Service/ClusterHealth"    rg\_name               = var.rg\_name    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Aks-CPU-Percentage" {    source                = "./Alerts/Azure\_Kubernetes\_Service/Aks-Cpu-Usage-Percentage"    rg\_name               = var.rg\_name    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Aks-Disk-Percentage" {    source                = "./Alerts/Azure\_Kubernetes\_Service/AKs-Disk-Usage-Percentage"    rg\_name               = var.rg\_name    action\_group\_id       = module.ActionGroup.action\_group\_id  }  module "Memory\_working\_set\_Percentage" {    source                = "./Alerts/Azure\_Kubernetes\_Service/Memory-WorkingSet\_Percentage"    rg\_name               = var.rg\_name    action\_group\_id       = module.ActionGroup.action\_group\_id  }  # module "ContainerCPU-Percentage" {  #   source                = "./Alerts/Azure\_Kubernetes\_Service/Container-CPU-Percentage"  #   rg\_name               = var.rg\_name  #   action\_group\_id       = module.ActionGroup.action\_group\_id  #   workspace\_resource\_id = module.law.log\_id    # }  # module "Container\_Memory\_Percentage" {  #   source                = "./Alerts/Azure\_Kubernetes\_Service/Container-Memory-Percentage"  #   rg\_name               = var.rg\_name  #   action\_group\_id       = module.ActionGroup.action\_group\_id  #   workspace\_resource\_id = module.law.log\_id    # }  # module "Average\_Node\_CPU" {  #   source                = "./Alerts/Azure\_Kubernetes\_Service/Average-Node-Cpu-Percentage"  #   rg\_name               = var.rg\_name  #   action\_group\_id       = module.ActionGroup.action\_group\_id  #   workspace\_resource\_id = module.law.log\_id    # }  # module "Readiness\_Status" {  #   source                = "./Alerts/Azure\_Kubernetes\_Service/Readiness status per node"  #   rg\_name               = var.rg\_name  #   action\_group\_id       = module.ActionGroup.action\_group\_id  #   workspace\_resource\_id = module.law.log\_id    # }  module "application\_gateway\_resource\_health" {    source  = "./Alerts/ApplicationGateway/Resource Health Alert"    rg\_name               = var.rg\_name    action\_group\_id       = module.ActionGroup.action\_group\_id    }  module "LogicApps" {     source = "./Alerts/Logic App/Actions Failed"     action\_group\_id = module.ActionGroup.action\_group\_id     rg\_name = var.rg\_name   }    module "LoadBalancer" {     source = "./Alerts/Load Balancer"     action\_group\_id = module.ActionGroup.action\_group\_id     rg\_name = var.rg\_name   }    module "Backup-ne" {    source = "./Alerts/BackupAlert - NE"    action\_group\_id = module.ActionGroup.action\_group\_id    rg\_name = var.rg\_name  }  module "Backup-ci" {    source = "./Alerts/BackupAlert - CI"    action\_group\_id = module.ActionGroup.action\_group\_id    rg\_name = var.rg\_name  } |

**Step 2:** Create a new folder with name Action group inside the subscription folder. Create two new files as mentioned below inside the Action Group folder.

* Actiongroup.tf - In this tf file, we are calling the action group created manually on portal via terraform data block or if not exists then create action group using resource block as shown below.
* Output.tf - prints the output
* Variable.tf- store the value of the variable

Here is the terraform code for actiongroup.tf

|  |
| --- |
| # data "azurerm\_monitor\_action\_group" "ag" {  #   name = "AG-AM-Dev-Subscription"  #   resource\_group\_name = "RG-AM-Dev-Subscription"  # }  resource "azurerm\_monitor\_action\_group" "ag" {    name                = var.action\_group\_name    resource\_group\_name = var.rg\_name    short\_name          = var.short\_name      dynamic "email\_receiver"{      for\_each = var.email\_reciever      content {        name = email\_receiver.value.email\_name        email\_address = email\_receiver.value.email\_address        use\_common\_alert\_schema = true    }  }  } |

Here is the code for output.tf

|  |
| --- |
| output "action\_group\_id" {      value = azurerm\_monitor\_action\_group.ag.id  } |
| Here is the code for variable.tf |
| variable "action\_group\_name" {    type = string    default = "AG-AM-Dev-Subscription"  }  variable "short\_name" {    type = string    default = "AG-AM-Dev"  }  variable "rg\_name" {    type = string    }  variable "email\_reciever" {    default = [      {      email\_address= "DBCInfraMgmt@hcltech.com",      email\_name = "DBCInfraMgmt"    },     {      email\_address= "DL-CLOUD-DB-CHEQUE@hcltech.com",      email\_name = "DL-Cloud-DB-CHEQUE"    }    ]  } |

**Step 3:** Create a new folder with name “LAW” inside the subscription folder**.** InsideLAW create three new files as mentioned below

* Law.tf - creating log analytics workspace by providing name of log analytics workspace and location and resource group under which its created.
* Variable.tf - Initializing the variable
* Output.tf - prints the output

Here is the terraform code for LAW.tf

|  |
| --- |
| resource "azurerm\_log\_analytics\_workspace" "law-Dev" {    name                = var.law\_name    location            = var.deploy\_location    resource\_group\_name = var.rg\_name    } |

Here is the terraform code for variable.tf

|  |
| --- |
| variable "law\_name" {    type = string    description = "Name of the LAW"    default = "LAW-AM-Dev-Subscription"  }  variable "deploy\_location" {    type        = string    description = "The Azure Region in which all resources in this example should be created."    default     = "North Europe"  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  } |

Here is the code for output.tf

|  |
| --- |
| output "log\_id"{    value =  azurerm\_log\_analytics\_workspace.law-Dev.id  } |

**Step4:** Create a new folder with name “DCR” inside the subscription. Inside DCR create two new files as mentioned below

* Dcr.tf - Data collection rule specify what data should be collected, and where to send that data.
* Variable.tf - Initializing the variable

Here is the terraform code for dcr.tf

|  |
| --- |
| locals {    names    = [for machine in var.virtual\_machines.vms : machine["vm\_name"]]    rg\_names = [for machine in var.virtual\_machines.vms : machine["resource\_group\_name"]]    os\_types = [for machine in var.virtual\_machines.vms : machine["os\_type"]]  }  data "azurerm\_virtual\_machine" "lab\_vm" {    count = length(local.names)    name                = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_data\_collection\_rule" "rule1" {    name                = var.dcr\_name    location            = var.deploy\_location    resource\_group\_name = var.rg\_name    destinations {      log\_analytics {        workspace\_resource\_id = var.workspace\_resource\_id        name                  = "log-analytics"      }      log\_analytics {        workspace\_resource\_id = var.workspace\_resource\_id        name                  = "log-analytics-event"      }      }    data\_flow {      streams      = ["Microsoft-Perf"]      destinations = ["log-analytics"]    }     data\_flow {      streams      = ["Microsoft-Event"]      destinations = ["log-analytics-event"]    }      data\_sources {      performance\_counter {        streams                       = ["Microsoft-Perf", "Microsoft-InsightsMetrics"]        sampling\_frequency\_in\_seconds = 60        counter\_specifiers = [          "\\Processor Information(\_Total)\\% Processor Time",          "\\Memory\\% Committed Bytes In Use",          "\\Processor(\_Total)\\% Processor Time",          "\\Memory\\% Used Memory",          "Memory\\Available Bytes",          "\\Memory\\% Used Swap Space",          "\\LogicalDisk(\*)\\% Free Space",          "\\Logical Disk(\*)\\% Used Space",           "Processor(\*)\\% Processor Time",          "\\LogicalDisk(\_Total)\\Disk Transfers/sec"            #   "Processor(\*)\\% Processor Time","\\Processor(\_Total)\\% Processor Time","\\LogicalDisk(\_Total)\\% Free Space","\\Memory\\% Committed Bytes In Use",          # "LogicalDisk(\*)\\% Free Space","Memory\\Available Bytes","Memory(\*)\\% Used Memory","Memory(\*)\\% Used Swap Space"        ]        name = "datasource-perfcounter"      }      windows\_event\_log {      streams = ["Microsoft-Event"]      x\_path\_queries = ["System!\*[System[(EventID=41 or EventID=1074 or EventID=6006 or EventID=6008)]]"]      name = "datasource-wineventlog"    }    }      }  resource "azurerm\_monitor\_data\_collection\_rule\_association" "dcra\_association" {    count                   = length(flatten(local.names))    name                    = "DCR-VM"    target\_resource\_id      = data.azurerm\_virtual\_machine.lab\_vm[count.index].id    data\_collection\_rule\_id = azurerm\_monitor\_data\_collection\_rule.rule1.id    description             = "Association between the Data Collection Rule and the Windows VM."  } |

Here is the terraform code for variable.tf

|  |
| --- |
| variable "workspace\_resource\_id" {  }  variable "dcr\_name" {    type = string    description = "Name of the DCR"    default = "DCR-AM-Dev-Subscription"  }  variable "deploy\_location" {    type        = string    default     = "North Europe"    description = "The Azure Region in which all resources in this example should be created."  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "virtual\_machines" {  } |

**Step 5:** Create a new folder with name Alerts inside subscription folder. Inside alerts folder create a folder as mentioned below

* + Linux Alerts
  + BackupAlert –Central India
  + BackupAlert-North Europe
  + Load Balancer
  + ApplicationGateway
  + Application Insight
  + VM Availability Alert-Central India
  + VM Availability Alert-North Europe
  + Window Alerts
  + Traffic Manager Profile
  + Logic App
  + Function App
  + Sql Managed Instance
  + PostgreSql Fexible server-North Europe
  + PostgreSql Fexible server-Central India
  + Data Factory
  + PostgresSql Single server
  + Azure Kurbnetes Services
* **LinuxAlerts** - Create four new folders as mentioned below inside the “LinuxAlerts” folder.
* CPU Utilization
* Logical Disk Used Space
* Memory Used Percentage
* Memory Used Swap Space
* **CPU Utilization –** CPU utilization is a measure of how much of the processor’s capacity is being used by the virtual machine. It is usually expressed as a percentage of the total CPU available on the virtual machine.
* Create two new files inside CPU utilization as mentioned below :
* **Variable.tf** – It is used to initializing the variables
* **cpu\_alert.tf** - Using the query to get the logs in LAW for virtual machines. Also taking, input threshold value, severity level, time aggregation method, operator from variable.tf file.

**Variable.tf**

|  |
| --- |
| variable "workspace\_resource\_id" {    type = string    description = "Workspace resource id to which logs will be sent"  }  variable "deploy\_location" {    type        = string    default = "North Europe"    description = "The Azure Region in which all resources in this example should be created."  }  variable "rg\_name" {    type = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {      type = string      default = "Linux Total CPU Utilization Percentage"  }  variable "threshold" {    type = list(number)    default = [85,90]    # default = [ 1,2 ]  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]  } |

**cpu\_alert.tf**

|  |
| --- |
| resource "azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2" "alert\_v2" {    count = length(var.threshold)    name                = "Alert-AM-Dev-Subscription-Linux-CpuUtilization-${var.severity\_level[count.index]}"    resource\_group\_name = var.rg\_name    location            = var.deploy\_location    evaluation\_frequency = "PT5M"    window\_duration      = "PT5M"    scopes               = [var.workspace\_resource\_id]    severity             = var.severity[count.index]    criteria {      query                   = <<-QUERY        Perf          | where ObjectName == "Processor" and CounterName == "% Processor Time" and InstanceName == "total"          | project TimeGenerated, Computer, CounterValue, \_ResourceId          QUERY      time\_aggregation\_method = var.time\_aggregation\_method      threshold               = var.threshold[count.index]      operator                = var.operator      resource\_id\_column    = "\_ResourceId"      metric\_measure\_column = "CounterValue"      dimension {        name     = "Computer"        operator = "Include"        values   = ["\*"]      }      failing\_periods {        minimum\_failing\_periods\_to\_trigger\_alert = 4        number\_of\_evaluation\_periods             = 4      }    }    auto\_mitigation\_enabled          = false    workspace\_alerts\_storage\_enabled = false    description                      = var.description    display\_name                     = "Alert-AM-Dev-Subscription-Linux-CpuUtilization-${var.severity\_level[count.index]}"    enabled                          = true    skip\_query\_validation            = false    action{    action\_groups = [var.action\_group\_id]  }   } |

* **LogicalDiskUsedSpace –** It indicates the amount of storage space currently utilized on the virtual machine’s logical disks.
* create two new files inside logical disk used space as mentioned below:
* logical\_disk\_used\_space\_alert.tf - Using the query to get the logs in LAW for virtual machines and getting input threshold value, severity level, time aggregation method, operator from variable.tf file.
* Variable.tf - It is used to initialize the variables.

**variable.tf:**

|  |
| --- |
| variable "workspace\_resource\_id" {  }  variable "deploy\_location" {    type        = string    default = "North Europe"    description = "The Azure Region in which all resources in this example should be created."  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Logical Disk UsedSpace"  }  variable "threshold" {    type = list(number)    # default = [0.0001,0.0002]    default = [ 85,90 ]  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]  } |

**logical\_disk\_used\_space\_alert:**

|  |
| --- |
| resource "azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2" "logical\_disk\_used\_space\_alert" {    count = length(var.threshold)    name                = "Alert-AM-Dev-Subscription-Linux-LogicalDiskUsedSpace-${var.severity\_level[count.index]}"    resource\_group\_name = var.rg\_name    location            = var.deploy\_location    evaluation\_frequency = "PT5M"    window\_duration      = "PT5M"    scopes               = [var.workspace\_resource\_id]    severity             = var.severity[count.index]    criteria {      query                   = <<-QUERY         Perf  | where ObjectName == 'Logical Disk' and CounterName == '% Used Space' and InstanceName !hasprefix "/snap" and InstanceName !hasprefix "/run" and InstanceName !hasprefix "/sys"  | summarize AggregatedValue = avg(CounterValue) by bin(TimeGenerated, 5m), Computer, InstanceName, \_ResourceId          QUERY      time\_aggregation\_method = var.time\_aggregation\_method      threshold               = var.threshold[count.index]      operator                = var.operator      resource\_id\_column    = "\_ResourceId"      metric\_measure\_column = "AggregatedValue"      dimension {        name     = "Computer"        operator = "Include"        values   = ["\*"]      }      dimension {        name     = "InstanceName"        operator = "Include"        values   = ["\*"]      }      failing\_periods {        minimum\_failing\_periods\_to\_trigger\_alert = 4        number\_of\_evaluation\_periods             = 4      }    }    auto\_mitigation\_enabled          = false    workspace\_alerts\_storage\_enabled = false    description                      = var.description    display\_name                     = "Alert-AM-Dev-Subscription-Linux-LogicalDiskUsedSpace-${var.severity\_level[count.index]}"    enabled                          = true    skip\_query\_validation            = false    action{    action\_groups = [var.action\_group\_id]  }  } |

* **Memory Used Percentage –** It represents the proportion of the available RAM that is currently being used by the system and its processes.
* Create two new files inside memory used percentage as mentioned below:
* memory\_used\_percentage\_alert.tf - Using the query to get the logs in LAW for virtual machines and getting input threshold value, severity level, time aggregation method, operator from variable.tf file.
* variable.tf – It is used to initialize the variables.

**variable.tf:**

|  |
| --- |
| variable "workspace\_resource\_id" {  }  variable "deploy\_location" {    type        = string    default = "North Europe"    description = "The Azure Region in which all resources in this example should be created."  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Memory Used percentage"  }  variable "threshold" {    type =  list(number)    default = [85,90]    # default = [4,5]  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]  } |

**memory\_used\_percentage\_alert.tf:**

|  |
| --- |
| resource "azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2" "alert\_v2" {    count = length(var.threshold)    name                = "Alert-AM-Dev-Subscription-Linux-MemoryUsed-${var.severity\_level[count.index]}"    resource\_group\_name = var.rg\_name    location            = var.deploy\_location    evaluation\_frequency = "PT5M"    window\_duration      = "PT5M"    scopes               = [var.workspace\_resource\_id]    severity             = var.severity[count.index]    criteria {      query                   = <<-QUERY        Perf          | where ObjectName == "Memory" and (CounterName == "% Used Memory")          | project CounterName,Computer,CounterValue,\_ResourceId,TimeGenerated          QUERY      time\_aggregation\_method = var.time\_aggregation\_method      threshold               = var.threshold[count.index]      operator                = var.operator      resource\_id\_column    = "\_ResourceId"      metric\_measure\_column = "CounterValue"      dimension {        name     = "Computer"        operator = "Include"        values   = ["\*"]      }      failing\_periods {        minimum\_failing\_periods\_to\_trigger\_alert = 4        number\_of\_evaluation\_periods             = 4      }    }    auto\_mitigation\_enabled          = false    workspace\_alerts\_storage\_enabled = false    description                      = var.description    display\_name                     = "Alert-AM-Dev-Subscription-Linux-MemoryUsed-${var.severity\_level[count.index]}"    enabled                          = true    skip\_query\_validation            = false    action{    action\_groups = [var.action\_group\_id]  }    } |

* **Memory Used Swap Space –** It refers to the portion of the system’s virtual memory that is being used in the swap space.
* create two new files inside memory used swap space as mentioned below
* memory\_used\_swap\_space\_alert.tf - - Using the query to get the logs in LAW for virtual machines and getting input threshold value, severity level, time aggregation method, operator from variable.tf file.
* variable.tf - It is used to initialize the variables.

**variable.tf:**

|  |
| --- |
| variable "workspace\_resource\_id" {  }  variable "deploy\_location" {    type        = string    default = "North Europe"    description = "The Azure Region in which all resources in this example should be created."  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Memory Used Swap Space"  }  variable "threshold" {    type = list(number)    default = [50,80]  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]  } |

**memory\_used\_swap\_space\_alert.tf:**

|  |
| --- |
| resource "azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2" "memory\_used\_swap\_space\_alert" {    count = length(var.threshold)    name                = "Alert-AM-Dev-Subscription-Linux-MemoryUsedSwapSpace-${var.severity\_level[count.index]}"    resource\_group\_name = var.rg\_name    location            = var.deploy\_location    evaluation\_frequency = "PT5M"    window\_duration      = "PT5M"    scopes               = [var.workspace\_resource\_id]    severity             = var.severity[count.index]    criteria {      query                   = <<-QUERY      Perf  | where ObjectName == "Memory" and CounterName == "% Used Swap Space"  | project TimeGenerated, Computer, CounterValue, \_ResourceId          QUERY      time\_aggregation\_method = var.time\_aggregation\_method      threshold               = var.threshold[count.index]      operator                = var.operator      resource\_id\_column    = "\_ResourceId"      metric\_measure\_column = "CounterValue"      dimension {        name     = "Computer"        operator = "Include"        values   = ["\*"]      }      failing\_periods {        minimum\_failing\_periods\_to\_trigger\_alert = 4        number\_of\_evaluation\_periods             = 4      }    }    auto\_mitigation\_enabled          = false    workspace\_alerts\_storage\_enabled = false    description                      = var.description    display\_name                     = "Alert-AM-Dev-Subscription-Linux-MemoryUsedSwapSpace-${var.severity\_level[count.index]}"    enabled                          = true    skip\_query\_validation            = false  action{    action\_groups = [var.action\_group\_id]  }      } |

* **Backup Alert –CI :** Adding the diagnostic settings in the recovery service vault and creating alert for failed backup job.
* create two new files inside Backup Alert as mentioned below :
* variable.tf – It is used to initialize the variables.
* backup.tf – Adding the diagnostic setting in the recovery service vault and creating alert for failed back up job.

**variable.tf:**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {      type = string      default = "The count of health events pertaining to backup job health"  }  variable "metric\_name" {    default = "BackupHealthEvent"  }  variable "threshold" {    type = number    default = 0  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Count"  }  variable "severity" {    type = number    default = 0  }  variable "recovery\_rg\_names" {    default = ["DEV-CI-AZBACKUP-001", "dev-ci-sql-paas-001"]  }  variable "recovery\_service\_vault\_names" {    default = ["RECSRVVault-DEV-CI-AZBACKUP-001", "vault-lsopnmx2"]  } |

**backup.tf:**

|  |
| --- |
| data "azurerm\_recovery\_services\_vault" "vault1" {    count = length(var.recovery\_service\_vault\_names)    name                = var.recovery\_service\_vault\_names[count.index]    resource\_group\_name = var.recovery\_rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "backup\_alert\_ci" {      name= "Alert-AM-Dev-Subscription-Backup-Failure-Critical-CentralIndia"      resource\_group\_name = var.rg\_name      scopes = toset(data.azurerm\_recovery\_services\_vault.vault1[\*].id)        severity = var.severity        frequency = "PT15M"      window\_size = "PT15M"      target\_resource\_location = "Central India"      target\_resource\_type = "Microsoft.RecoveryServices/vaults"        criteria {      metric\_namespace = "Microsoft.RecoveryServices/vaults"      metric\_name      = var.metric\_name      aggregation      = var.time\_aggregation\_method      operator         = var.operator      threshold        = var.threshold      dimension{         name = "healthStatus"         operator = "Include"         values = ["PersistentDegraded","TransientDegraded","TransientUnhealthy"]      }      dimension{         name = "backupInstanceName"         operator = "Include"         values = ["\*"]      }      }         action {      action\_group\_id = var.action\_group\_id    }      } |

* **Backup Alert –NE:** Adding the diagnostic settings in the recovery service vault and creating alert for failed backup job.
* create two new files inside Backup Alert as mentioned below :
* variable.tf – It is used to initialize the variables.
* backup.tf – Adding the diagnostic setting in the recovery service vault and creating alert for failed back up job.

**variable.tf:**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {      type = string      default = "The count of health events pertaining to backup job health"  }  variable "metric\_name" {    default = "BackupHealthEvent"  }  variable "threshold" {    type = number    default = 0  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Count"  }  variable "severity" {    type = number    default = 0  }  variable "recovery\_rg\_names" {    default = ["DEV-NE-AZBACKUP-001", "DEV-NE-APACIL-RG-001"]  }  variable "recovery\_service\_vault\_names" {    default = ["RECSRVVault-DEV-NE-AZBACKUP-001", "vault-lts96nm6"]  } |

**backup.tf:**

|  |
| --- |
| data "azurerm\_recovery\_services\_vault" "vault1" {    count = length(var.recovery\_service\_vault\_names)    name                = var.recovery\_service\_vault\_names[count.index]    resource\_group\_name = var.recovery\_rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "backup\_alert-ne" {      name= "Alert-AM-Dev-Subscription-Backup-Failure-Critical-NorthEurope"      resource\_group\_name = var.rg\_name      scopes = toset(data.azurerm\_recovery\_services\_vault.vault1[\*].id)        severity = var.severity        frequency = "PT15M"      window\_size = "PT15M"      target\_resource\_location = "North Europe"      target\_resource\_type = "Microsoft.RecoveryServices/vaults"        criteria {      metric\_namespace = "Microsoft.RecoveryServices/vaults"      metric\_name      = var.metric\_name      aggregation      = var.time\_aggregation\_method      operator         = var.operator      threshold        = var.threshold      dimension{         name = "healthStatus"         operator = "Include"         values = ["PersistentDegraded","TransientDegraded","TransientUnhealthy"]      }      dimension{         name = "backupInstanceName"         operator = "Include"         values = ["\*"]      }      }         action {      action\_group\_id = var.action\_group\_id    }      } |

* **Load Balancer:**
* **Dip Availability :** This metric measures the average load balancer health probe status per time duration.
* Create two new files inside Load Balancer as mentioned below
* Variable.tf – It is used to initializing the variables
* DipAvailability.tf - using metric alert to monitor DipAvailability by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {      type = string      default = "Average Load Balancer health probe status per time duration"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Count"  }  variable "metric\_name" {    default = "DipAvailability"  }  variable "metric\_alerts" {    type = map(object({      lb\_name = string      rg\_name = string      severity = number      threshold = number    }))    default = {      kubernetes-MC\_DEV-NE-AKS-RG-GB-001\_devneakscluster001\_northeurope-Error = {        lb\_name = "kubernetes"        rg\_name = "MC\_DEV-NE-AKS-RG-GB-001\_devneakscluster001\_northeurope"        severity = 1        threshold = 2      },      kubernetes-MC\_DEV-NE-AKS-RG-GB-001\_devneakscluster001\_northeurope-Warning = {        lb\_name = "kubernetes"        rg\_name = "MC\_DEV-NE-AKS-RG-GB-001\_devneakscluster001\_northeurope"        severity = 2        threshold = 1      },      kubernetes-mc\_dev-ci-aks-rg-gb-001\_devciakscluster001\_centralindia-Error = {        lb\_name = "kubernetes"        rg\_name = "mc\_dev-ci-aks-rg-gb-001\_devciakscluster001\_centralindia"        severity = 1        threshold = 2      },      kubernetes-mc\_dev-ci-aks-rg-gb-001\_devciakscluster001\_centralindia-Warning = {        lb\_name = "kubernetes"        rg\_name = "mc\_dev-ci-aks-rg-gb-001\_devciakscluster001\_centralindia"        severity = 2        threshold = 1      },      kubernetes-internal-mc\_dev-ci-aks-rg-gb-001\_devciakscluster001\_centralindia-Error = {        lb\_name = "kubernetes-internal"        rg\_name = "mc\_dev-ci-aks-rg-gb-001\_devciakscluster001\_centralindia"        severity = 1        threshold = 2      },      kubernetes-internal-mc\_dev-ci-aks-rg-gb-001\_devciakscluster001\_centralindia-Warning = {        lb\_name = "kubernetes-internal"        rg\_name = "mc\_dev-ci-aks-rg-gb-001\_devciakscluster001\_centralindia"        severity = 2        threshold = 1      },      kubernetes-internal-mc\_dev-ne-aks-rg-gb-001\_devneakscluster001\_northeurope-Error = {        lb\_name = "kubernetes-internal"        rg\_name = "mc\_dev-ne-aks-rg-gb-001\_devneakscluster001\_northeurope"        severity = 1        threshold = 2      },      kubernetes-internal-mc\_dev-ne-aks-rg-gb-001\_devneakscluster001\_northeurope-Warning = {        lb\_name = "kubernetes-internal"        rg\_name = "mc\_dev-ne-aks-rg-gb-001\_devneakscluster001\_northeurope"        severity = 2        threshold = 1      }      }  } |

**DipAvailability.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "load\_balancer\_alert" {    for\_each = var.metric\_alerts    name = "Alert-AM-Dev-Subscription-LoadBalancer-${each.key}-HealthProbeStatus"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/a86c54d3-8722-41d7-aebe-6b1106f49431/resourceGroups/${each.value.rg\_name}/providers/Microsoft.Network/loadBalancers/${each.value.lb\_name}"]    description = var.description    severity = each.value.severity    enabled = true  criteria {    metric\_namespace = "Microsoft.Network/loadBalancers"    metric\_name = var.metric\_name    aggregation = var.time\_aggregation\_method    operator = var.operator    threshold = each.value.threshold  }   action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* **VM Availability-CI:** Allows you to track the pulse of your VMs—during expected behavior, the metric displays a value of 1. In response to any VM availability disruptions, the metric dips to a 0 for the duration of impact.
* Create a new folder with name “VMAvailabilityAlert” inside the Alerts folder. Inside the newly created folder, create two new files as mentioned below:
* VMAvailabilityAlert.tf : using metric alert to monitor VM Availability  by providing the metric\_namespace, threshold, frequency, window\_size.
* Variable.tf : It is used to initializing the variables.

**VMAvailabilityAlert.tf**

|  |
| --- |
| locals {    names = [for machine in var.virtual\_machines\_centralIndia.vms : machine["vm\_name"]]    rg\_names = [for machine in var.virtual\_machines\_centralIndia.vms : machine["resource\_group\_name"]]  }  data "azurerm\_virtual\_machine" "lab\_vm" {  count = length(local.names)   name = local.names[count.index]   resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "VM-rule-availability-Ci" {    name = "Alert-AM-Dev-Subscription-VMAvailability-Critical-CentralIndia"    resource\_group\_name = var.rg\_name    description         = "The metric allows you to track the pulse of your VMs—during expected behavior, the metric displays a value of 1. In response to any VM availability disruptions, the metric dips to a 0 for the duration of impact. "    scopes              = toset(data.azurerm\_virtual\_machine.lab\_vm[\*].id)    target\_resource\_type = "Microsoft.Compute/virtualMachines"    target\_resource\_location = "Central India"    severity = var.severity    criteria {      metric\_namespace = "Microsoft.Compute/virtualMachines"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "The percentage of availability for the virtual machines"  }  variable "threshold" {    type = number    default = 1  }  variable "operator" {      type = string    default = "LessThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = number    default = 0  }  variable "metric\_name"{      type = string      default = "VmAvailabilityMetric"  }  variable "virtual\_machines\_centralIndia"{    type = map(any)    default = {      vms = [{       vm\_name ="devciaksjump002",        resource\_group\_name ="dev-ci-aks-rg-gb-001",        os\_type = "linux"      },      {       vm\_name ="DEVCICTS00001",        resource\_group\_name ="dev-ci-cts-rg-001",        os\_type = "windows"      },        {       vm\_name ="DEVCICTS02",        resource\_group\_name ="dev-ci-cts-rg-001",        os\_type = "windows"      },      {       vm\_name ="devcidboracle1",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "linux"      },      {       vm\_name ="devcidboracle2",        resource\_group\_name ="dev-ci-gis-rg-002",        os\_type = "linux"      },        {       vm\_name ="DEVCIDEVIAC002",        resource\_group\_name ="dev-ci-iac-rg-001",        os\_type = "linux"      },        {       vm\_name ="devcielk00001",        resource\_group\_name ="dev-ci-apacil-rg-001",        os\_type = "linux"      },        {       vm\_name ="devciexacto0001",        resource\_group\_name ="dev-ci-iclear-rg-001",        os\_type = "linux"      },      {       vm\_name ="devciexacto0002",        resource\_group\_name ="dev-ci-iclear-rg-001",        os\_type = "linux"      },       {       vm\_name ="devcigis00001",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {       vm\_name ="devcihwa00001",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "linux"      },       {       vm\_name ="DEVCIICLCTWIN01",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "windows"      },       {       vm\_name ="devcijasper0001",        resource\_group\_name ="dev-ci-iclear-rg-001",        os\_type = "linux"      },       {       vm\_name ="devcijenkins01",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "linux"      },       {       vm\_name ="DEVCISFTP00001",        resource\_group\_name ="dev-ci-apacil-rg-001",        os\_type = "windows"      },       {       vm\_name ="devcisftp11",        resource\_group\_name ="dev-ci-apacil-rg-001",        os\_type = "linux"      },       {       vm\_name ="Precigis00001",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },       {       vm\_name ="preciiclct001",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },       {       vm\_name ="preciiclct002",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },       {       vm\_name ="PRFTSTCIAPSCN1",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },       {       vm\_name ="PRFTSTCIAPSCN2",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },       {       vm\_name ="rubrikpocaks-vm-0",        resource\_group\_name ="dev-ci-rubrik-rg-001",        os\_type = "linux"      },       {       vm\_name ="SITCIICHQ00001",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },       {       vm\_name ="sitciiclct001",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },       {       vm\_name ="sitciiclct002",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      },       {       vm\_name ="UATCIGISAUTO004",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {       vm\_name ="UATCIGISAUTO005",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {       vm\_name ="UATCIGISAUTO006",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {       vm\_name ="uatciiclct001",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "linux"      }      ]    }  } |

* **VM Availability-NE:** Allows you to track the pulse of your VMs—during expected behavior, the metric displays a value of 1. In response to any VM availability disruptions, the metric dips to a 0 for the duration of impact.
* Create a new folder with name “VMAvailabilityAlert” inside the Alerts folder. Inside the newly created folder, create two new files as mentioned below:
* VMAvailabilityAlert.tf : using metric alert to monitor VM Availability  by providing the metric\_namespace, threshold, frequency, window\_size.
* Variable.tf : It is used to initializing the variables.

**VMAvailabilityAlert.tf**

|  |
| --- |
| locals {    names = [for machine in var.virtual\_machines\_northeurope.vms : machine["vm\_name"]]    rg\_names = [for machine in var.virtual\_machines\_northeurope.vms : machine["resource\_group\_name"]]  }  data "azurerm\_virtual\_machine" "lab\_vm" {  count = length(local.names)   name = local.names[count.index]   resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "VM-rule-availability-ne" {    name = "Alert-AM-Dev-Subscription-VMAvailability-Critical-NorthEurope"    resource\_group\_name = var.rg\_name    description         = "The metric allows you to track the pulse of your VMs—during expected behavior, the metric displays a value of 1. In response to any VM availability disruptions, the metric dips to a 0 for the duration of impact. "    scopes              = toset(data.azurerm\_virtual\_machine.lab\_vm[\*].id)    target\_resource\_type = "Microsoft.Compute/virtualMachines"    target\_resource\_location = "North Europe"    severity = var.severity    criteria {      metric\_namespace = "Microsoft.Compute/virtualMachines"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "The percentage of availability for the virtual machines"  }  variable "threshold" {    type = number    default = 1  }  variable "operator" {      type = string    default = "LessThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = number    default = 0  }  variable "metric\_name"{      type = string      default = "VmAvailabilityMetric"  }  variable "virtual\_machines\_northeurope"{    type = map(any)    default = {      vms = [{       vm\_name ="devneaksjump002",        resource\_group\_name ="dev-ne-aks-rg-gb-001",        os\_type = "linux"      },      {       vm\_name ="devnechk00001",        resource\_group\_name ="dev-ne-chk-rg-001",        os\_type = "linux"      },      {       vm\_name ="devneelk00001",        resource\_group\_name ="dev-ne-usil-rg-001",        os\_type = "linux"      },      {       vm\_name ="DEVNEIDM00001",        resource\_group\_name ="dev-ne-idmcre-rg-001",        os\_type = "windows"      },        {       vm\_name ="DEVNESFTP00001",        resource\_group\_name ="dev-ne-apacil-rg-001",        os\_type = "windows"      },      {       vm\_name ="devnesftp11",        resource\_group\_name ="dev-ne-apacil-rg-001",        os\_type = "linux"      },      {       vm\_name ="SITNEICHQ00001",        resource\_group\_name ="dev-ne-hwa-rg-001",        os\_type = "windows"      },       {       vm\_name ="UATNEIDM00001",        resource\_group\_name ="dev-ne-idmcre-rg-001",        os\_type = "windows"      }      ]    }  } |

* **Windows Alerts:**

Create four new folders as mentioned below inside the “WindowsAlerts” folder.

* + - * CPU Utilization
      * DataDiskIOPS
      * Logical Disk Free Space
      * Memory Available Bytes
      * System Event Log (Win Shutdown)
* **CPU Utilization Alert:** This metric is the average percent CPU used during the period specified as the data collection interval.
* Create two new files inside CPU utilization as mentioned below:
* Variable.tf – It is used to initializing the variables
* cpu\_alert.tf - Using the query to get the logs in LAW for virtual machines. Also taking, input threshold value, severity level, time aggregation method, operator from variable.tf file.

**variable.tf**

|  |
| --- |
| variable "workspace\_resource\_id" {    description = "Workspace resource id to which logs will be sent"  }  variable "deploy\_location" {    type        = string    description = "The Azure Region in which all resources in this example should be created."    default = "North Europe"  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {      type = string      default = "Total CPU Utilization Percentage"  }  variable "threshold" {    type = list(number)    default = [85,90]    # default = [1,2]  }  variable "operator" {    type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]  } |

**CPUUtilization.tf**

|  |
| --- |
| resource "azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2" "cpu-alert-windows" {      count = length(var.threshold)    name                = "Alert-AM-Dev-Subscription-Windows-CpuUtilization-${var.severity\_level[count.index]}"    resource\_group\_name = var.rg\_name    location            = var.deploy\_location    evaluation\_frequency = "PT5M"    window\_duration      = "PT5M"    scopes               = [var.workspace\_resource\_id]    severity             = var.severity[count.index]    criteria {      query                   = <<-QUERY        Perf          | where ObjectName == "Processor Information" and CounterName == "% Processor Time" and InstanceName == "\_Total"          | project TimeGenerated, Computer, CounterValue, \_ResourceId          QUERY      time\_aggregation\_method = var.time\_aggregation\_method      threshold               = var.threshold[count.index]      operator                = var.operator      resource\_id\_column    = "\_ResourceId"      metric\_measure\_column = "CounterValue"      dimension {        name     = "Computer"        operator = "Include"        values   = ["\*"]      }      failing\_periods {        minimum\_failing\_periods\_to\_trigger\_alert = 4        number\_of\_evaluation\_periods             = 4      }    }    auto\_mitigation\_enabled          = false    workspace\_alerts\_storage\_enabled = false    description                      = var.description    display\_name                     = "Alert-AM-Dev-Subscription-Windows-CpuUtilization-${var.severity\_level[count.index]}"    enabled                          = true    skip\_query\_validation            = false    action{    action\_groups = [var.action\_group\_id]  }  } |

* **DataDiskIOPS – Central India:** It refers to the percentage calculated by the data disk IOPS completed over the provisioned data disk IOPS. If this amount is at 100%, your application running is IO capped from your data disk's IOPS limit.

Create two new files inside DataDiskIOPS as mentioned below :

* Variable.tf – It is used to initializing the variables.
* DataDiskIOPS.tf -  In this file, taking input threshold value, severity level, time aggregation method, operator from variable.tf file.

**Variable.tf**

|  |
| --- |
| variable "severity" {    type = list(number)    default = [2,1]  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {      type = string      default = "Percentage of data disk I/Os consumed per minute"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "virtual\_machines\_centralIndia" {    type = map(any)    default = {      vms = [{       vm\_name ="DEVCICTS00001",        resource\_group\_name ="dev-ci-cts-rg-001",        os\_type = "windows"      },        {       vm\_name ="DEVCICTS02",        resource\_group\_name ="dev-ci-cts-rg-001",        os\_type = "windows"      },      {       vm\_name ="devcigis00001",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },        {       vm\_name ="DEVCIICLCTWIN01",        resource\_group\_name ="dev-ci-icollect-rg-01",        os\_type = "windows"      },      {       vm\_name ="DEVCISFTP00001",        resource\_group\_name ="dev-ci-apacil-rg-001",        os\_type = "windows"      },       {       vm\_name ="Precigis00001",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },       {       vm\_name ="PRFTSTCIAPSCN1",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },      {       vm\_name ="PRFTSTCIAPSCN2",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },      {       vm\_name ="SITCIICHQ00001",        resource\_group\_name ="dev-ci-hwa-rg-001",        os\_type = "windows"      },      {       vm\_name ="UATCIGISAUTO004",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {       vm\_name ="UATCIGISAUTO005",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      },      {       vm\_name ="UATCIGISAUTO006",        resource\_group\_name ="dev-ci-gis-rg-001",        os\_type = "windows"      }      ]    }  }  variable "metric\_name" {    default = "Data Disk IOPS Consumed Percentage"  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]    }  variable "threshold" {    type = list(number)    default = [85,90]    # default = [ 1,2 ]  } |

**DataDiskIOPS.tf**

|  |
| --- |
| locals {    names = [for machine in var.virtual\_machines\_centralIndia.vms : machine["vm\_name"]]    rg\_names = [for machine in var.virtual\_machines\_centralIndia.vms : machine["resource\_group\_name"]]    os\_types = [for machine in var.virtual\_machines\_centralIndia.vms : machine["os\_type"]]  }  data "azurerm\_virtual\_machine" "lab\_vm" {   count = length(local.names)   name = local.names[count.index]   resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "data\_disk\_alert\_dev-ci" {    count = length(var.severity\_level)    name = "Alert-AM-Dev-Subscription-Windows-DataDiskIOPS-${var.severity\_level[count.index]}-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_virtual\_machine.lab\_vm[\*].id)    description = var.description    severity = var.severity[count.index]    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.Compute/virtualMachines"    criteria {      metric\_namespace = "Microsoft.Compute/virtualMachines"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold[count.index]    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* **DataDiskIOPS-North Europe :** It refers to the percentage calculated by the data disk IOPS completed over the provisioned data disk IOPS. If this amount is at 100%, your application running is IO capped from your data disk's IOPS limit.
* Create two new files inside DataDiskIOPS as mentioned below :
* Variable.tf – It is used to initializing the variables.
* DataDiskIOPS.tf -  In this file, taking input threshold value, severity level, time aggregation method, operator from variable.tf file.

**Variable.tf**

|  |
| --- |
| variable "severity" {    type = list(number)    default = [2,1]  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {      type = string      default = "Percentage of data disk I/Os consumed per minute"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "virtual\_machines\_northeurope" {    type = map(any)    default = {      vms = [{       vm\_name ="DEVNEIDM00001",        resource\_group\_name ="dev-ne-idmcre-rg-001",        os\_type = "windows"      },        {       vm\_name ="DEVNESFTP00001",        resource\_group\_name ="dev-ne-apacil-rg-001",        os\_type = "windows"      },      {       vm\_name ="SITNEICHQ00001",        resource\_group\_name ="dev-ne-hwa-rg-001",        os\_type = "windows"      },        {       vm\_name ="UATNEIDM00001",        resource\_group\_name ="dev-ne-idmcre-rg-001",        os\_type = "windows"      }        ]    }  }  variable "metric\_name" {    default = "Data Disk IOPS Consumed Percentage"  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]    }  variable "threshold" {    type = list(number)    default = [85,90]    # default = [ 1,2 ]  } |

**DataDiskIOPS.tf**

|  |
| --- |
| locals {    names = [for machine in var.virtual\_machines\_northeurope.vms : machine["vm\_name"]]    rg\_names = [for machine in var.virtual\_machines\_northeurope.vms : machine["resource\_group\_name"]]    os\_types = [for machine in var.virtual\_machines\_northeurope.vms : machine["os\_type"]]  }  data "azurerm\_virtual\_machine" "lab\_vm" {   count = length(local.names)   name = local.names[count.index]   resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "data\_disk\_alert\_dev-ne" {    count = length(var.severity\_level)    name = "Alert-AM-Dev-Subscription-Windows-DataDiskIOPS-${var.severity\_level[count.index]}-NorthEurope"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_virtual\_machine.lab\_vm[\*].id)    description = var.description    severity = var.severity[count.index]    target\_resource\_location = "North Europe"    target\_resource\_type = "Microsoft.Compute/virtualMachines"    criteria {      metric\_namespace = "Microsoft.Compute/virtualMachines"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold[count.index]    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

**Variable.tf –Logical Disk Free Space**

|  |
| --- |
| variable "workspace\_resource\_id" {    description = "Workspace resource id to which lohs will be sent"  }  variable "deploy\_location" {    type        = string    default = "North Europe"    description = "The Azure Region in which all resources in this example should be created."  }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {    type = string    default = "Logical Disk Free Space"  }  variable "threshold" {    type = list(number)    default = [15,10]    # default = [90,95]  }  variable "operator" {      type = string    default = "LessThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]  } |

* **Logical Disk Free Space:** It refers to the amount of available storage space on the logical disks of the virtual machine. This metric provides information about the remaining free space on the disks where your operating system, applications, and data are stored.
* Create two new files inside LogicalDiskFreeSpace as mentioned below:
* Variable.tf – It is used to initializing the variables
* logicaldiskfreespace.tf - Using the query to get the logs in LAW for virtual machines. Also taking, input threshold value, severity level, time aggregation method, operator from variable.tf file.

**LogicalDiskFreeSpace.tf**

|  |
| --- |
| resource "azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2" "logical\_disk\_free\_space\_alert" {    count = length(var.threshold)    name                = "Alert-AM-Dev-Subscription-Windows-LogicalDiskFreeSpace-${var.severity\_level[count.index]}"    resource\_group\_name = var.rg\_name    location            = var.deploy\_location    evaluation\_frequency = "PT5M"    window\_duration      = "PT5M"    scopes               = [var.workspace\_resource\_id]    severity             = var.severity[count.index]    criteria {      query                   = <<-QUERY        Perf | where ObjectName == 'LogicalDisk' and CounterName == '% Free Space' and InstanceName != '\_Total' and InstanceName != 'D:' and InstanceName !startswith 'HarddiskVolume' | summarize AggregatedValue = avg(CounterValue) by bin(TimeGenerated, 5m), Computer, InstanceName,\_ResourceId          QUERY      time\_aggregation\_method = var.time\_aggregation\_method      threshold               = var.threshold[count.index]      operator                = var.operator      resource\_id\_column    = "\_ResourceId"      metric\_measure\_column = "AggregatedValue"      dimension {        name     = "Computer"        operator = "Include"        values   = ["\*"]      }      failing\_periods {        minimum\_failing\_periods\_to\_trigger\_alert = 4        number\_of\_evaluation\_periods             = 4      }    }    auto\_mitigation\_enabled          = false    workspace\_alerts\_storage\_enabled = false    description                      = var.description    display\_name                     = "Alert-AM-Dev-Subscription-Windows-LogicalDiskFreeSpace-${var.severity\_level[count.index]}"    enabled                          = true    skip\_query\_validation            = false    action{    action\_groups = [var.action\_group\_id]  }    } |

* **MemoryAvailableBytes :** It represents the amount of RAM (Random Access Memory) that is currently unused and available for use by the system and applications on the virtual machine. This metric is a measure of the free memory that can be utilized for running processes.
* Create two new files inside MemoryAvailableBytes as mentioned below:
* Variable.tf – It is used to initializing the variables
* MemoryAvailableBytes.tf - Using the query to get the logs in LAW for virtual machines. Also taking, input threshold value, severity level, time aggregation method, operator from variable.tf file.

**Variable.tf**

|  |
| --- |
| variable "workspace\_resource\_id" {    }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    }  variable "description" {      type = string      default = "Memory Available Megabytes"  }  variable "deploy\_location" {    type        = string    description = "The Azure Region in which all resources in this example should be created."    default = "North Europe"  }  variable "threshold" {    type = list(number)    default = [85,90]  #  default = [5,6]  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "severity\_level" {      type = list(string)      default = ["Warning","Error"]  } |

**MemoryAvailableBytes.tf**

|  |
| --- |
| resource "azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2" "memory\_available\_bytes\_alert" {     count = length(var.threshold)    name                = "Alert-AM-Dev-Subscription-Windows-MemoryAvailable-${var.severity\_level[count.index]}"    resource\_group\_name = var.rg\_name    location            = var.deploy\_location    evaluation\_frequency = "PT5M"    window\_duration      = "PT5M"    scopes               = [var.workspace\_resource\_id]    severity             = var.severity[count.index]    criteria {      query                   = <<-QUERY       Perf        | where ObjectName == "Memory" and (CounterName == "% Committed Bytes In Use")        | project CounterName,Computer,CounterValue,\_ResourceId,TimeGenerated          QUERY      time\_aggregation\_method = var.time\_aggregation\_method         threshold               = var.threshold[count.index]      operator                = var.operator      resource\_id\_column    = "\_ResourceId"      metric\_measure\_column = "CounterValue"      dimension {        name     = "Computer"        operator = "Include"        values   = ["\*"]      }      failing\_periods {        minimum\_failing\_periods\_to\_trigger\_alert = 4        number\_of\_evaluation\_periods             = 4      }    }    auto\_mitigation\_enabled          = false    workspace\_alerts\_storage\_enabled = false    description                      = var.description    display\_name                     = "Alert-AM-Dev-Subscription-Windows-MemoryAvailable-${var.severity\_level[count.index]}"    enabled                          = true   # query\_time\_range\_override        = "P2D"    skip\_query\_validation            = false   action{    action\_groups = [var.action\_group\_id]  }  } |

* **System Event Log Shutdown Start:** It refers to entries in the Windows Event Log that document the startup and shutdown events of the virtual machine. This log contains information about when the Azure VM was started or stopped.
* Create two new files inside SystemEventLog as mentioned below:
* Variable.tf – It is used to initializing the variables
* shutdownstart.tf - Using the query to get the logs in LAW for virtual machines. Also taking, input threshold value, severity level, time aggregation method, operator from variable.tf file.

**Variable.tf**

|  |
| --- |
| variable "workspace\_resource\_id" {    }  variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    }  variable "description" {    type = string    default = "Windows Event ID 41, 1074, 6006, 6008 / 6005"  }  variable "deploy\_location" {    type        = string    description = "The Azure Region in which all resources in this example should be created."    default = "North Europe"  }  variable "time\_aggregation\_method" {    type = string    default = "Count"  } |

**Shutdownstart.tf**

|  |
| --- |
| resource "azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2" "startstopalert\_v2" {    name                = "Alert-AM-Dev-Subscription-Windows-Shutdown-Critical"    resource\_group\_name = var.rg\_name    location            = var.deploy\_location    evaluation\_frequency = "PT5M"    window\_duration      = "PT5M"    scopes               = [var.workspace\_resource\_id]    severity             = 0    criteria {      query                   = <<-QUERY        Event        | where EventID in (41, 1074, 6006, 6008) and \_ResourceId != ""        | project TimeGenerated, Computer, EventID, RenderedDescription,\_ResourceId         QUERY       time\_aggregation\_method = "Count"         threshold               = 0       operator                = "GreaterThan"        resource\_id\_column    = "\_ResourceId"        dimension {        name     = "Computer"        operator = "Include"        values   = ["\*"]      }      dimension {        name     = "EventID"        operator = "Include"        values   = ["\*"]      }      dimension {        name     = "RenderedDescription"        operator = "Include"        values   = ["\*"]      }      failing\_periods {        minimum\_failing\_periods\_to\_trigger\_alert = 1        number\_of\_evaluation\_periods             = 1      }    }      auto\_mitigation\_enabled          = false    workspace\_alerts\_storage\_enabled = false    description                      = var.description    display\_name                     = "Alert-AM-Dev-Subscription-Windows-Shutdown-Critical"    enabled                          = true    skip\_query\_validation            = false    action {      action\_groups =  [var.action\_group\_id]      }    } |

* **Application Gateway**: Create five new folders as mentioned below:
* ResourceHealth Alert
* **Resource Health Alert :** Resource Health on anApplication Gateway relies on signals emitted by the gateway to assess whether it’s healthy or not.
* Create two new files inside Resource Health Alert as mentioned below:
* Variable.tf – It is used to initializing the variables
* Resourcehealth.tf -   using activity log alerts to monitor Resource Health alert by mentioning category, operation name and resource type

**Variable.tf**

|  |
| --- |
| variable "action\_group\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "category" {      type = string    default = "ResourceHealth"  }  variable "description" {      type = string    default = "Resource Health Alert"  }  variable "app\_gateway\_name" {    default = ["devneakscluster-appgateway","ingress-appgateway","ingress-appgateway"]  }  variable "app\_gateway\_rg\_name" {    default = ["MC\_DEV-NE-AKS-RG-GB-001\_devneakscluster001\_northeurope","MC\_DEV-CI-AKS-RG-GB-001\_devciakscluster001\_centralindia","MC\_DEV-NE-AKS-RG-GB-001\_devneakscluster001\_northeurope"]  } |

**Resource\_health.tf**

|  |
| --- |
| data "azurerm\_application\_gateway" "application\_gateway" {    count = length(var.app\_gateway\_name)    name  = var.app\_gateway\_name[count.index]    resource\_group\_name = var.app\_gateway\_rg\_name[count.index]  }  resource "azurerm\_monitor\_activity\_log\_alert" "resource\_health\_alert" {    name                = "Alert-AM-Dev-Subscription-AppGateway-ResourceHealth"    resource\_group\_name = var.rg\_name    scopes              = ["/subscriptions/a86c54d3-8722-41d7-aebe-6b1106f49431"]    description         = var.description      criteria {      resource\_ids    = toset(data.azurerm\_application\_gateway.application\_gateway[\*].id)       resource\_type  = "Microsoft.Network/applicationGateways"      resource\_groups = toset(data.azurerm\_application\_gateway.application\_gateway[\*].resource\_group\_name)      category       = var.category         resource\_health {           current = ["Degraded","Unavailable"]           previous = ["Available"]           reason = ["PlatformInitiated","UserInitiated","Unknown"]      }    }    action {      action\_group\_id = var.action\_group\_id    }    } |

* **Application Insight**: Create eight new folders under Application Insight as mentioned below:
  + - Availability
    - Browser Exceptions
    - Browser Page Load Time
    - CPU
    - Dependency Duration
    - Failed Requests
    - Server Response Time
    - Server Exceptions
* **Availability Alert:** The Availability metric shows the percentage of the web test runs that didn't detect any issues. The lowest possible value is 0, which indicates that all the web test runs have failed. The value of 100 means that all the web test runs passed the validation criteria.

Create two new files inside Availability Alert as mentioned below:

* Variable.tf – It is used to initializing the variables**.**
* Availabilty\_alert.tf -   using metric alert to monitor availability of app insight by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Availability Alert"  }  variable "metric\_name" {      type = string    default = "availabilityResults/availabilityPercentage"  }  variable "operator" {      type = string    default = "LessThan"  }  variable "aggregation" {    type = string    default = "Average"  }  variable "metric\_alert" {    type = map(object({      insight\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {      dev-ci-gis-001-Warning= {       insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 2        threshold = 95      }      dev-ci-gis-001-Error= {         insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 1        threshold = 85      }      iclear-g2b-function-Warning= {       insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 95      }      iclear-g2b-function-Error= {        insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 85      }      iclear-g2b-function-dryrun-Warning= {       insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 95      }      iclear-g2b-function-dryrun-Error= {        insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 85      }      icollect-apac-preprod-Warning= {       insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 2        threshold = 95      }      icollect-apac-preprod-Error= {        insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 1        threshold = 85      }      Postgres-Job-Schedule-Warning= {       insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 95      }      Postgres-Job-Schedule-Error= {        insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 1        threshold = 85      }      prd-ci-iclear-function-001-Warning= {       insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 95      }      prd-ci-iclear-function-001-Error= {        insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 85      }      pre-ci-gis-001-Warning= {       insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 2        threshold = 95      }      pre-ci-gis-001-Error = {        insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 1        threshold = 85      }      prod-ci-iclear-g2b-function-Warning= {       insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 95      }      prod-ci-iclear-g2b-function-Error= {        insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 85      }      testfunction1241-Warning= {       insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 2        threshold = 95      }      testfunction1241-Error= {        insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 1        threshold = 85      }      uat-ci-iclear-function-001-Warning= {       insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 95      }      uat-ci-iclear-function-001-Error= {        insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 85      }      usdisb-eu-preprod-Warning= {       insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 2        threshold = 95      }      usdisb-eu-preprod-Error= {         insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 1        threshold = 85      }    }  } |

**Availability\_alert.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "AppInsightavailability\_alert" {    for\_each = var.metric\_alert    name = "Alert-AM-Hub-Subscription-AppInsight-Availability-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.insights/components/${each.value.insight\_name}"]    description = var.description    severity = each.value.severity    criteria {      metric\_namespace = "microsoft.insights/components"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* **Browser Exceptions:** This metric reflects the number of thrown exceptions from your application code running in browser. Only exceptions that are tracked with a trackException() Application Insights API call are included in the metric.Create two new files inside Browser Exceptions folder as mentioned below:
* Variable.tf – It is used to initializing the variables.
* BrowserExceptions.tf -   using metric alert to monitor availability of app insight by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Browser page load time"  }  variable "metric\_name" {      type = string    default = "exceptions/browser"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Count"  }  variable "metric\_alert" {    type = map(object({      insight\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {      dev-ci-gis-001-Warning= {       insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 2        threshold = 400      }      dev-ci-gis-001-Error= {         insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 1        threshold = 600      }      iclear-g2b-function-Warning= {       insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 400      }      iclear-g2b-function-Error= {        insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 600      }      iclear-g2b-function-dryrun-Warning= {       insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 400      }      iclear-g2b-function-dryrun-Error= {        insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 600      }      icollect-apac-preprod-Warning= {       insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 2        threshold = 400      }      icollect-apac-preprod-Error= {        insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 1        threshold = 600      }      Postgres-Job-Schedule-Warning= {       insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 400      }      Postgres-Job-Schedule-Error= {        insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 1        threshold = 600      }      prd-ci-iclear-function-001-Warning= {       insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 400      }      prd-ci-iclear-function-001-Error= {        insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 600      }      pre-ci-gis-001-Warning= {       insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 2        threshold = 400      }      pre-ci-gis-001-Error = {        insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 1        threshold = 600      }      prod-ci-iclear-g2b-function-Warning= {       insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 400      }      prod-ci-iclear-g2b-function-Error= {        insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 600      }      testfunction1241-Warning= {       insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 2        threshold = 400      }      testfunction1241-Error= {        insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 1        threshold = 600      }      uat-ci-iclear-function-001-Warning= {       insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 400      }      uat-ci-iclear-function-001-Error= {        insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 600      }      usdisb-eu-preprod-Warning= {       insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 2        threshold = 400      }      usdisb-eu-preprod-Error= {         insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 1        threshold = 600      }    }  } |

**Browserexceptions.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "browser\_exceptions\_alert" {    for\_each = var.metric\_alert    name =  "Alert-AM-Hub-Subscription-AppInsight-BrowserExceptions-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.insights/components/${each.value.insight\_name}"]    severity = each.value.severity     description = var.description    criteria {      metric\_namespace = "microsoft.insights/components"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT30M"    frequency = "PT30M"  } |

* **Browser Page Load Time:**  Time from user request until DOM, stylesheets, scripts and images are loaded.
* Create two new files inside Browser Page Load Time folder as mentioned below:
* Variable.tf – It is **used to initializing the variables.**
* browserpageloadtime.tf -   using metric alert to monitor browser page load time of app insight by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Browser page load time Alert"  }  variable "metric\_name" {      type = string    default = "browserTimings/totalDuration"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Average"  }  variable "metric\_alert" {    type = map(object({      insight\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {      dev-ci-gis-001-Warning= {       insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 2        threshold = 3000      }      dev-ci-gis-001-Error= {         insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 1        threshold = 5000      }      iclear-g2b-function-Warning= {       insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 3000      }      iclear-g2b-function-Error= {        insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 5000      }      iclear-g2b-function-dryrun-Warning= {       insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 3000      }      iclear-g2b-function-dryrun-Error= {        insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 5000      }      icollect-apac-preprod-Warning= {       insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 2        threshold = 3000      }      icollect-apac-preprod-Error= {        insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 1        threshold = 5000      }      Postgres-Job-Schedule-Warning= {       insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 3000      }      Postgres-Job-Schedule-Error= {        insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 1        threshold = 5000      }      prd-ci-iclear-function-001-Warning= {       insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 3000      }      prd-ci-iclear-function-001-Error= {        insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 5000      }      pre-ci-gis-001-Warning= {       insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 2        threshold = 3000      }      pre-ci-gis-001-Error = {        insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 1        threshold = 5000      }      prod-ci-iclear-g2b-function-Warning= {       insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 3000      }      prod-ci-iclear-g2b-function-Error= {        insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 5000      }      testfunction1241-Warning= {       insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 2        threshold = 3000      }      testfunction1241-Error= {        insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 1        threshold = 5000      }      uat-ci-iclear-function-001-Warning= {       insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 3000      }      uat-ci-iclear-function-001-Error= {        insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 5000      }      usdisb-eu-preprod-Warning= {       insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 2        threshold = 3000      }      usdisb-eu-preprod-Error= {         insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 1        threshold = 5000      }    }  } |

**Browserpageloadtime.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "browser\_page\_load\_time\_alert" {    for\_each = var.metric\_alert    name =  "Alert-AM-Hub-Subscription-AppInsight-BrowserPageLoadTime-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.insights/components/${each.value.insight\_name}"]    severity = each.value.severity     description = var.description    criteria {      metric\_namespace = "microsoft.insights/components"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* **CPU Alert:** The metric shows how much of the total processor capacity is consumed by the process that is hosting your monitored app.
* Create two new files inside CPU Alert folder as mentioned below:
* Variable.tf – It is used to initializing the variables.
* cpu\_alert.tf -   using metric alert to monitor availability of app insight by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    }  variable "description" {      type = string      default = "Process CPU Alert"    }  variable "metric\_name" {      type = string    default = "performanceCounters/processCpuPercentage"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Average"  }  variable "metric\_alert" {    type = map(object({      insight\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {      dev-ci-gis-001-Warning= {       insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 2        threshold = 80      }      dev-ci-gis-001-Error= {         insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 1        threshold = 90      }      iclear-g2b-function-Warning= {       insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 80      }      iclear-g2b-function-Error= {        insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 90      }      iclear-g2b-function-dryrun-Warning= {       insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 80      }      iclear-g2b-function-dryrun-Error= {        insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 90      }      icollect-apac-preprod-Warning= {       insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 2        threshold = 80      }      icollect-apac-preprod-Error= {        insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 1        threshold = 90      }      Postgres-Job-Schedule-Warning= {       insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 80      }      Postgres-Job-Schedule-Error= {        insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 1        threshold = 90      }      prd-ci-iclear-function-001-Warning= {       insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 80      }      prd-ci-iclear-function-001-Error= {        insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 90      }      pre-ci-gis-001-Warning= {       insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 2        threshold = 80      }      pre-ci-gis-001-Error = {        insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 1        threshold = 90      }      prod-ci-iclear-g2b-function-Warning= {       insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 80      }      prod-ci-iclear-g2b-function-Error= {        insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 90      }      testfunction1241-Warning= {       insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 2        threshold = 80      }      testfunction1241-Error= {        insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 1        threshold = 90      }      uat-ci-iclear-function-001-Warning= {       insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 80      }      uat-ci-iclear-function-001-Error= {        insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 90      }      usdisb-eu-preprod-Warning= {       insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 2        threshold = 80      }      usdisb-eu-preprod-Error= {         insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 1        threshold = 90      }    }  } |

**Cpu\_alert.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "AppInsight\_cpu\_alert" {    for\_each = var.metric\_alert    name = "Alert-AM-Hub-Subscription-AppInsight-ProcessCPU-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.insights/components/${each.value.insight\_name}"]    description = var.description    severity = each.value.severity    criteria {      metric\_namespace = "microsoft.insights/components"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* **Dependency Duration :** This metric refers to the duration of dependency calls.

Create two new files inside Dependency Duration folder as mentioned below:

* Variable.tf – It is used to initializing the **variables.**

1. dependencyduration.tf -   using metric alert to monitor dependency duration of app insight by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Client processing time"  }  variable "metric\_name" {      type = string    default = "dependencies/duration"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Average"  }  variable "metric\_alert" {    type = map(object({      insight\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {      dev-ci-gis-001-Warning= {       insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 2        threshold = 3000      }      dev-ci-gis-001-Error= {         insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 1        threshold = 5000      }      iclear-g2b-function-Warning= {       insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 3000      }      iclear-g2b-function-Error= {        insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 5000      }      iclear-g2b-function-dryrun-Warning= {       insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 3000      }      iclear-g2b-function-dryrun-Error= {        insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 5000      }      icollect-apac-preprod-Warning= {       insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 2        threshold = 3000      }      icollect-apac-preprod-Error= {        insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 1        threshold = 5000      }      Postgres-Job-Schedule-Warning= {       insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 3000      }      Postgres-Job-Schedule-Error= {        insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 1        threshold = 5000      }      prd-ci-iclear-function-001-Warning= {       insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 3000      }      prd-ci-iclear-function-001-Error= {        insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 5000      }      pre-ci-gis-001-Warning= {       insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 2        threshold = 3000      }      pre-ci-gis-001-Error = {        insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 1        threshold = 5000      }      prod-ci-iclear-g2b-function-Warning= {       insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 3000      }      prod-ci-iclear-g2b-function-Error= {        insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 5000      }      testfunction1241-Warning= {       insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 2        threshold = 3000      }      testfunction1241-Error= {        insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 1        threshold = 5000      }      uat-ci-iclear-function-001-Warning= {       insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 3000      }      uat-ci-iclear-function-001-Error= {        insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 5000      }      usdisb-eu-preprod-Warning= {       insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 2        threshold = 3000      }      usdisb-eu-preprod-Error= {         insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 1        threshold = 5000      }    }  } |

**Dependencyduration.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "dependency\_durations\_alert" {    for\_each = var.metric\_alert    name =  "Alert-AM-Hub-Subscription-AppInsight-DependencyDurations-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.insights/components/${each.value.insight\_name}"]    severity = each.value.severity    description = var.description    criteria {      metric\_namespace = "microsoft.insights/components"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold      }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* **Failed Requests :** The count of tracked server requests that were marked as failed. By default, the Application Insights SDK automatically marks each server request that returned HTTP response code 5xx or 4xx as a failed request.
* Create two new files inside FailedRequests folder as mentioned below:
* Variable.tf – It is used to initializing the variables.
* FailedRequests.tf -   using metric alert to monitor availability of app insight by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Failed requests"  }  variable "metric\_name" {      type = string    default = "requests/failed"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Count"  }  variable "metric\_alert" {    type = map(object({      insight\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {      dev-ci-gis-001-Warning= {       insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 2        threshold = 200      }      dev-ci-gis-001-Error= {         insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 1        threshold = 250      }      iclear-g2b-function-Warning= {       insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 200      }      iclear-g2b-function-Error= {        insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 250      }      iclear-g2b-function-dryrun-Warning= {       insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 200      }      iclear-g2b-function-dryrun-Error= {        insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 250      }      icollect-apac-preprod-Warning= {       insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 2        threshold = 200      }      icollect-apac-preprod-Error= {        insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 1        threshold = 250      }      Postgres-Job-Schedule-Warning= {       insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 200      }      Postgres-Job-Schedule-Error= {        insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 1        threshold = 250      }      prd-ci-iclear-function-001-Warning= {       insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 200      }      prd-ci-iclear-function-001-Error= {        insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 250      }      pre-ci-gis-001-Warning= {       insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 2        threshold = 200      }      pre-ci-gis-001-Error = {        insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 1        threshold = 250      }      prod-ci-iclear-g2b-function-Warning= {       insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 200      }      prod-ci-iclear-g2b-function-Error= {        insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 250      }      testfunction1241-Warning= {       insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 2        threshold = 200      }      testfunction1241-Error= {        insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 1        threshold = 250      }      uat-ci-iclear-function-001-Warning= {       insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 200      }      uat-ci-iclear-function-001-Error= {        insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 250      }      usdisb-eu-preprod-Warning= {       insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 2        threshold = 200      }      usdisb-eu-preprod-Error= {         insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 1        threshold = 250      }    }  } |

**FailedRequests.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "failedrequests\_alert" {    for\_each = var.metric\_alert    name =  "Alert-AM-Hub-Subscription-AppInsight-FailedRequests-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.insights/components/${each.value.insight\_name}"]    severity = each.value.severity    description = var.description    criteria {      metric\_namespace = "microsoft.insights/components"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* **Server Response Time :** This metric reflects the time it took for the servers to process incoming requests.

Create two new files inside Server Response Time folder as mentioned below:

* Variable.tf – It is **used to initializing the variables.**

1. Response\_time\_alert.tf -   using metric alert to monitor availability of app insight by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    }  variable "description" {      type = string      default = "Server response time"  }  variable "metric\_name" {      type = string    default = "requests/duration"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Average"  }  variable "metric\_alert" {    type = map(object({      insight\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {      dev-ci-gis-001-Warning= {       insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 2        threshold = 2000      }      dev-ci-gis-001-Error= {         insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 1        threshold = 4000      }      iclear-g2b-function-Warning= {       insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 2000      }      iclear-g2b-function-Error= {        insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 4000      }      iclear-g2b-function-dryrun-Warning= {       insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 2000      }      iclear-g2b-function-dryrun-Error= {        insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 4000      }      icollect-apac-preprod-Warning= {       insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 2        threshold = 2000      }      icollect-apac-preprod-Error= {        insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 1        threshold = 4000      }      Postgres-Job-Schedule-Warning= {       insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 2000      }      Postgres-Job-Schedule-Error= {        insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 1        threshold = 4000      }      prd-ci-iclear-function-001-Warning= {       insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 2000      }      prd-ci-iclear-function-001-Error= {        insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 4000      }      pre-ci-gis-001-Warning= {       insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 2        threshold = 2000      }      pre-ci-gis-001-Error = {        insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 1        threshold = 4000      }      prod-ci-iclear-g2b-function-Warning= {       insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 2000      }      prod-ci-iclear-g2b-function-Error= {        insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 4000      }      testfunction1241-Warning= {       insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 2        threshold = 2000      }      testfunction1241-Error= {        insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 1        threshold = 4000      }      uat-ci-iclear-function-001-Warning= {       insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 2000      }      uat-ci-iclear-function-001-Error= {        insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 4000      }      usdisb-eu-preprod-Warning= {       insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 2        threshold = 2000      }      usdisb-eu-preprod-Error= {         insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 1        threshold = 4000      }    }  } |

**Response\_time\_alert.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "responsetime\_alert" {    for\_each = var.metric\_alert    name =  "Alert-AM-Hub-Subscription-AppInsight-ServerReponseTime-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.insights/components/${each.value.insight\_name}"]    severity = each.value.severity    description = var.description    criteria {      metric\_namespace = "microsoft.insights/components"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* **Server Exceptions:** This metric shows the number of server exceptions.

Create two new files inside Server Exceptions folder as mentioned below:

* 1. Variable.tf – It is used to initializing the **variables.**
  2. Serverexceptions.tf -   using metric alert to monitor server exception of app insight by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    }  variable "description" {      type = string      default = "Server exceptions"  }  variable "metric\_name" {      type = string    default = "exceptions/server"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Count"  }  variable "metric\_alert" {    type = map(object({      insight\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {      dev-ci-gis-001-Warning= {       insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 2        threshold = 250      }      dev-ci-gis-001-Error= {         insight\_name = "dev-ci-gis-001"        rg\_name = "DEV-CI-GIS-RG-001"        severity = 1        threshold = 400      }      iclear-g2b-function-Warning= {       insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 250      }      iclear-g2b-function-Error= {        insight\_name = "iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 400      }      iclear-g2b-function-dryrun-Warning= {       insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 250      }      iclear-g2b-function-dryrun-Error= {        insight\_name = "iclear-g2b-function-dryrun"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 400      }      icollect-apac-preprod-Warning= {       insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 2        threshold = 250      }      icollect-apac-preprod-Error= {        insight\_name = "icollect-apac-preprod"        rg\_name = "DEV-CI-ICOLLECT-RG-001"        severity = 1        threshold = 400      }      Postgres-Job-Schedule-Warning= {       insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 250      }      Postgres-Job-Schedule-Error= {        insight\_name = "Postgres-Job-Schedule"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 1        threshold = 400      }      prd-ci-iclear-function-001-Warning= {       insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 250      }      prd-ci-iclear-function-001-Error= {        insight\_name = "prd-ci-iclear-function-001"        rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 400      }      pre-ci-gis-001-Warning= {       insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 2        threshold = 250      }      pre-ci-gis-001-Error = {        insight\_name = "pre-ci-gis-001"        rg\_name = "PRE-CI-AKS-001"        severity = 1        threshold = 400      }      prod-ci-iclear-g2b-function-Warning= {       insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 250      }      prod-ci-iclear-g2b-function-Error= {        insight\_name = "prod-ci-iclear-g2b-function"        rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 400      }      testfunction1241-Warning= {       insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 2        threshold = 250      }      testfunction1241-Error= {        insight\_name = "testfunction1241"        rg\_name = "testfunction1241"        severity = 1        threshold = 400      }      uat-ci-iclear-function-001-Warning= {       insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 250      }      uat-ci-iclear-function-001-Error= {        insight\_name = "uat-ci-iclear-function-001"        rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 400      }      usdisb-eu-preprod-Warning= {       insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 2        threshold = 250      }      usdisb-eu-preprod-Error= {         insight\_name = "usdisb-eu-preprod"        rg\_name = "DEV-NE-AKS-RG-001"        severity = 1        threshold = 400      }    }  } |

**Serverexceptions.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "serverexceptions\_alert" {    for\_each = var.metric\_alert    name =  "Alert-AM-Hub-Subscription-AppInsight-ServerExceptions-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.insights/components/${each.value.insight\_name}"]    severity = each.value.severity    description = var.description    criteria {      metric\_namespace = "microsoft.insights/components"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* **Traffic Manager Profile**
* Create a folder as mentioned below under Traffic Manager Profile:
* variable.tf - It is used to initialize the variables.
* Resource\_health.tf - using metric alert to monitor Resourcehealth by providing the metric\_namespace, threshold, frequency, window\_size.
* **Resource Health -** Monitors the health of the resource. The alert will be fired if the health status is unavailable or degraded.

Create two new files inside Resource Health folder as mentioned below:

**Variable.tf**

|  |
| --- |
| variable "action\_group\_id" {    description = "Action Group Id"  }  variable "rg\_name" {  }  variable "category" {      type = string    default = "ResourceHealth"  }  variable "description" {      type = string    default = "Resource Health Alert"  }  variable "trafficmanagerprofile" {     default = ["Citrix-apac","CTS","cts-hclchequesutility-com","CTS-UAT","epcidm-eu","epcidm-eu-hclchequesutility-com","gis-apac-india-hclchequesutility-com","iclearing-apac","iclearing-apac-hclchequesutility-com","icollect-apac-callback-hclchequesutility-com","icollect-apac-hclchequesutility-com","SFTP-APAC","SFTP-APAC-TEST","SFTP-DEV-APAC","SFTP-DEV-US","SFTP-POC","SFTP-TM","SFTP-TM-TEST","usdisb-eu","usdisb-eu-hclchequesutility-com"]   }  variable "trafficmanagerrgname" {     default = ["HUB-CI-CITRIX-RG-001","PRD-CI-CTS-RG-001","PRD-CI-CTS-RG-001","DEV-CI-CTS-RG-001","PD-NE-SFTP-RG-001","PD-NE-SFTP-RG-001","PRD-CI-GIS-RG-001","HUB-CI-CITRIX-RG-001","PD-CI-iClear-RG-001","PD-CI-ICOLLECT-RG-01","PRD-CI-CTS-RG-001","pd-ci-sftp-rg","PD-CI-SFTP-RG","hub-ci-sftp-nrg-001","HUB-NE-SFTP-NRG-001","POC-WE-SFTP-RG-001","pd-ne-sftp-rg-001","PD-NE-SFTP-RG-001","PD-NE-USDB-RG-001","PD-NE-USDB-RG-001"]   } |

**resource\_health.tf**

|  |
| --- |
| data "azurerm\_traffic\_manager\_profile" "tfmanagerprofile" {    count = length(var.trafficmanagerprofile)    name                = var.trafficmanagerprofile[count.index]    resource\_group\_name = var.trafficmanagerrgname[count.index]  }  resource "azurerm\_monitor\_activity\_log\_alert" "traffic\_manager\_profile\_resource\_health" {    name                = "Alert-AM-Hub-Subscription-TrafficManagerProfile-ResourceHealth"    resource\_group\_name = var.rg\_name    scopes              = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a"]    description         = var.description     criteria {       resource\_type  = "Microsoft.Network/trafficManagerProfiles"      category       = var.category      resource\_groups = toset(data.azurerm\_traffic\_manager\_profile.tfmanagerprofile[\*].resource\_group\_name)      resource\_ids = toset(data.azurerm\_traffic\_manager\_profile.tfmanagerprofile[\*].id)         resource\_health {           current = ["Degraded","Unavailable"]           previous = ["Available"]           reason = ["PlatformInitiated","UserInitiated","Unknown"]      }    }    action {      action\_group\_id = var.action\_group\_id    }  } |

* **Logic App**

Create four new folders under Logic App as mentioned below:

* + - Actions Failed
* **Actions Failed** - Actions Failed refers to instances where one or more actions within the logic app workflow encounter an issue or fail to execute successfully.

Create two new files inside Logic App folder as mentioned below:

* Variable.tf – It is used to initialize the variables.
* Actions\_failed.tf -   using metric alert to monitor actions failed of Logic App by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    description = "Action Group Id"  }  variable "description" {    type    = string    default = "Number of workflow actions failed."  }  variable "operator" {    type    = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type    = string    default = "Count"  }  variable "metric\_name" {    default = "ActionsFailed"  }  variable "severity" {    default = 0  }  variable "threshold" {    default = 0  }  variable "metric\_alert" {    type = map(object({      LogicApp\_name = string      rg\_name = string    }))    default = {      LA-DEV-CI-APACIL-KV-02 = {        LogicApp\_name = "LA-DEV-CI-APACIL-KV-02"        rg\_name = "DEV-CI-KV-EXP-NOT-001"        },      LA-DEV-CI-CTS-DB-KV-02 = {        LogicApp\_name = "LA-DEV-CI-CTS-DB-KV-02"        rg\_name = "DEV-CI-KV-EXP-NOT-001"        },      LA-DEV-CI-CTS-KV-02 = {        LogicApp\_name = "LA-DEV-CI-CTS-KV-02"        rg\_name = "DEV-CI-KV-EXP-NOT-001"        },      LA-DEV-CI-EDEPOSIT-KV-02 = {        LogicApp\_name = "LA-DEV-CI-EDEPOSIT-KV-02"        rg\_name = "DEV-CI-KV-EXP-NOT-001"        },      LA-DEV-CI-EDP-DB-KV-02 = {        LogicApp\_name = "LA-DEV-CI-EDP-DB-KV-02"        rg\_name = "DEV-CI-KV-EXP-NOT-001"        }    }  } |

**Actions\_failed.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "DBC-logic" {    for\_each = var.metric\_alert    name                     = "Alert-AM-Dev-Subscription-LogicApp-${each.key}-ActionsFailed-Critical"    resource\_group\_name      = var.rg\_name    scopes                   = ["/subscriptions/a86c54d3-8722-41d7-aebe-6b1106f49431/resourceGroups/${each.value.rg\_name}/providers/Microsoft.Logic/workflows/${each.value.LogicApp\_name}"]    severity                 = var.severity    description = "Logic App Alert"    criteria {      metric\_namespace = "Microsoft.Logic/workflows"      metric\_name      = var.metric\_name      aggregation      = var.time\_aggregation\_method      operator         = var.operator      threshold        = var.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* Function App
* Create three new folders under Logic App as mentioned below:
* Http 5xx
* Http Response Time
* **Http 5xx** - Http 5xx error indicates when the function app was unable to complete the execution of a function due to an internal server error.
* Create two new files inside Http 5xx folder as mentioned below:
* Variable.tf – It is used to initialize the variables.
* Http\_5xx.tf -   using metric alert to monitor Http 5xx errors of Function App by providing the metric\_namespace, threshold, frequency, window\_size.

**Vairabe.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    }  variable "description" {      type = string      default = "The count of requests resulting in an HTTP status code = 500 but < 600. "  }  variable "display\_name" {      type = string    default = "Http Server Errors"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "metric\_name"{      type = string      default = "Http5xx"  }  variable "metric\_alerts" {    default = {      arcsight-monitor-functionapp-Warning = {        functionApp\_name = "arcsight-monitor-functionapp"        functionApp\_rg\_name = "arcsightfunctionsgroup"        severity = 2        threshold = 50      },      arcsight-monitor-functionapp-Error = {        functionApp\_name = "arcsight-monitor-functionapp"        functionApp\_rg\_name = "arcsightfunctionsgroup"        severity = 1        threshold = 100      },      arcsightcloudfunctionapp-Warning = {        functionApp\_name = "arcsightcloudfunctionapp"        functionApp\_rg\_name = "arcsightfunctionsgroup"        severity = 2        threshold = 50      },      arcsightcloudfunctionapp-Error = {        functionApp\_name = "arcsightcloudfunctionapp"        functionApp\_rg\_name = "arcsightfunctionsgroup"        severity = 1        threshold = 100      },      iclear-g2b-function-dryrun-Warning = {        functionApp\_name = "iclear-g2b-function-dryrun"        functionApp\_rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 50      },      iclear-g2b-function-dryrun-Error = {        functionApp\_name = "iclear-g2b-function-dryrun"        functionApp\_rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 100      },      prd-ci-iclear-function-001-Warning = {        functionApp\_name = "prd-ci-iclear-function-001"        functionApp\_rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 50      },      prd-ci-iclear-function-001-Error = {        functionApp\_name = "prd-ci-iclear-function-001"        functionApp\_rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 100      },      prd-si-iclear-function-001-Warning = {        functionApp\_name = "prd-si-iclear-function-001"        functionApp\_rg\_name = "PD-SI-iCLEAR-RG-001"        severity = 2        threshold = 50      },      prd-si-iclear-function-001-Error = {        functionApp\_name = "prd-si-iclear-function-001"        functionApp\_rg\_name = "PD-SI-iCLEAR-RG-001"        severity = 1        threshold = 100      },      testfunction1241-Warning = {        functionApp\_name = "testfunction1241"        functionApp\_rg\_name = "testfunction1241"        severity = 2        threshold = 50      },      testfunction1241-Error = {        functionApp\_name = "testfunction1241"        functionApp\_rg\_name = "testfunction1241"        severity = 1        threshold = 100      },      uat-ci-iclear-function-001-Warning = {        functionApp\_name = "uat-ci-iclear-function-001"        functionApp\_rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 50      },      uat-ci-iclear-function-001-Error = {        functionApp\_name = "uat-ci-iclear-function-001"        functionApp\_rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 100      }      }  } |

**Http\_5xx.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "function\_app\_http5xx" {    for\_each = var.metric\_alerts    name = "Alert-AM-Hub-Subscription-FunctionApp-Http5xx-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.functionApp\_rg\_name}/providers/Microsoft.Web/sites/${each.value.functionApp\_name}"]    description = var.description    severity = each.value.severity    criteria {      metric\_namespace = "microsoft.web/sites"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }     window\_size = "PT15M"    frequency = "PT15M"  } |

* **Http Response time** - The response time of the function app is the time it takes for the function to complete its execution whenever triggered.
* Create two new files inside Response Time folder as mentioned below:
* Variable.tf – It is used to initializethe variables.
* Response\_time.tf -   using metric alert to monitor response time of Function App by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {    }  variable "description" {      type = string      default = "The time taken for the app to serve requests, in seconds. For WebApps and FunctionApps."  }  variable "display\_name" {      type = string    default = "Response Time"  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "aggregation" {    type = string    default = "Average"  }  variable "severity" {    type = list(number)    default = [2,1]  }  variable "metric\_name"{      type = string      default = "HttpResponseTime"  }  variable "metric\_alerts" {    default = {      arcsight-monitor-functionapp-Warning = {        functionApp\_name = "arcsight-monitor-functionapp"        functionApp\_rg\_name = "arcsightfunctionsgroup"        severity = 2        threshold = 10      },      arcsight-monitor-functionapp-Error = {        functionApp\_name = "arcsight-monitor-functionapp"        functionApp\_rg\_name = "arcsightfunctionsgroup"        severity = 1        threshold = 15      },      arcsightcloudfunctionapp-Warning = {        functionApp\_name = "arcsightcloudfunctionapp"        functionApp\_rg\_name = "arcsightfunctionsgroup"        severity = 2        threshold = 10      },      arcsightcloudfunctionapp-Error = {        functionApp\_name = "arcsightcloudfunctionapp"        functionApp\_rg\_name = "arcsightfunctionsgroup"        severity = 1        threshold = 15      },      iclear-g2b-function-dryrun-Warning = {        functionApp\_name = "iclear-g2b-function-dryrun"        functionApp\_rg\_name = "iclear-G2B-function-RG"        severity = 2        threshold = 10      },      iclear-g2b-function-dryrun-Error = {        functionApp\_name = "iclear-g2b-function-dryrun"        functionApp\_rg\_name = "iclear-G2B-function-RG"        severity = 1        threshold = 15      },      prd-ci-iclear-function-001-Warning = {        functionApp\_name = "prd-ci-iclear-function-001"        functionApp\_rg\_name = "PD-CI-iClear-RG-001"        severity = 2        threshold = 10      },      prd-ci-iclear-function-001-Error = {        functionApp\_name = "prd-ci-iclear-function-001"        functionApp\_rg\_name = "PD-CI-iClear-RG-001"        severity = 1        threshold = 15      },      prd-si-iclear-function-001-Warning = {        functionApp\_name = "prd-si-iclear-function-001"        functionApp\_rg\_name = "PD-SI-iCLEAR-RG-001"        severity = 2        threshold = 10      },      prd-si-iclear-function-001-Error = {        functionApp\_name = "prd-si-iclear-function-001"        functionApp\_rg\_name = "PD-SI-iCLEAR-RG-001"        severity = 1        threshold = 15      },      testfunction1241-Warning = {        functionApp\_name = "testfunction1241"        functionApp\_rg\_name = "testfunction1241"        severity = 2        threshold = 10      },      testfunction1241-Error = {        functionApp\_name = "testfunction1241"        functionApp\_rg\_name = "testfunction1241"        severity = 1        threshold = 15      },      uat-ci-iclear-function-001-Warning = {        functionApp\_name = "uat-ci-iclear-function-001"        functionApp\_rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 2        threshold = 10      },      uat-ci-iclear-function-001-Error = {        functionApp\_name = "uat-ci-iclear-function-001"        functionApp\_rg\_name = "DEV-CI-iCLEAR-RG-001"        severity = 1        threshold = 15      }      }  } |

**HTTP Response\_time.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "function\_app\_response\_time\_alert" {    for\_each = var.metric\_alerts    name = "Alert-AM-Hub-Subscription-FunctionApp-HTTPResponseTime-${each.key}"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.functionApp\_rg\_name}/providers/Microsoft.Web/sites/${each.value.functionApp\_name}"]    description = var.description    severity = each.value.severity    criteria {      metric\_namespace = "microsoft.web/sites"      metric\_name = var.metric\_name      aggregation = var.aggregation      operator = var.operator      threshold = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* **Sql Managed Instance** – for this, you need to create a sql managed instance folder under which two-difffernt sub folder should create. One for Average CPU and second one is Storage Space usage in SQL Managed Instances.
* **Average CPU Percent**- This alert will trigger when cpu uage/utilization of sql managed instance is go beyound defined threshold , then alert will trigger for cpu usage

Create two new files inside **Average CPU Percent** as mentioned below:

* Variable.tf – It is used to initializethe variables.
* Avg\_cpu\_perecent.tf -   using metric to trigger the alert by providing the metric\_namespace, threshold, frequency, window\_size.

* **Storage Space Usage** - This alert will trigger when storage utilization of sql managed instance is go beyound defined threshold, then alert will trigger for storage usage.

Create two new files inside **Storage Space Usage** as mentioned below:

* Variable.tf – It is used to initializethe variables.
* Storage\_space\_used.tf -   using metric to trigger the alert when storage space used exceeds its value by providing the metric\_namespace, threshold, frequency, window\_size.

**Variable.tf**

|  |
| --- |
| variable "action\_group\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "avg\_cpu\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "metric\_alert" {      type = map(object({      sql\_managedInstance\_name = string      rg\_name = string      severity = number      threshold = number      }))      default = {      devcimssqlcts001-Warning = {        sql\_managedInstance\_name = "devcimssqlcts001"        rg\_name = "dev-ci-sql-paas-001"        severity = 2        threshold = 85        #threshold = 2      }        devcimssqlcts001-Error = {        sql\_managedInstance\_name = "devcimssqlcts001"        rg\_name = "dev-ci-sql-paas-001"        severity = 1        threshold = 95        #threshold = 1      }      devcimssqliclearuat001-Warning = {        sql\_managedInstance\_name = "devcimssqliclearuat001"        rg\_name = "dev-ci-sql-paas-001"        severity = 2        threshold = 85        #threshold = 2      }        devcimssqliclearuat001-Error = {        sql\_managedInstance\_name = "devcimssqliclearuat001"        rg\_name = "dev-ci-sql-paas-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqlcts001-Warning = {        sql\_managedInstance\_name = "pdcimssqlcts001"        rg\_name = "pd-ci-sqlcltr-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqlcts001-Error = {        sql\_managedInstance\_name = "pdcimssqlcts001"        rg\_name = "pd-ci-sqlcltr-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqledeposit001-Warning = {        sql\_managedInstance\_name = "pdcimssqledeposit001"        rg\_name = "pd-ci-sqlmi-edeposit-nrg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqledeposit001-Error = {        sql\_managedInstance\_name = "pdcimssqledeposit001"        rg\_name = "pd-ci-sqlmi-edeposit-nrg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqliclear001-Warning = {        sql\_managedInstance\_name = "pdcimssqliclear001"        rg\_name = "pd-ci-sqlcltr-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqliclear001-Error = {        sql\_managedInstance\_name = "pdcimssqliclear001"        rg\_name = "pd-ci-sqlcltr-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqliclearhk002-Warning = {        sql\_managedInstance\_name = "pdcimssqliclearhk002"        rg\_name = "PD-CI-SQLCLTR-RG-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqliclearhk002-Error = {        sql\_managedInstance\_name = "pdcimssqliclearhk002"        rg\_name = "PD-CI-SQLCLTR-RG-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqlil001-Warning = {        sql\_managedInstance\_name = "pdcimssqlil001"        rg\_name = "pd-ci-il-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqlil001-Error = {        sql\_managedInstance\_name = "pdcimssqlil001"        rg\_name = "pd-ci-il-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdsimssqlcts001-Warning = {        sql\_managedInstance\_name = "pdsimssqlcts001"        rg\_name = "pd-si-sqlcltr-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqlcts001-Error = {        sql\_managedInstance\_name = "pdsimssqlcts001"        rg\_name = "pd-si-sqlcltr-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdsimssqledeposit001-Warning = {        sql\_managedInstance\_name = "pdsimssqledeposit001"        rg\_name = "pd-si-sqlmi-edeposit-nrg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqledeposit001-Error = {        sql\_managedInstance\_name = "pdsimssqledeposit001"        rg\_name = "pd-si-sqlmi-edeposit-nrg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdsimssqliclear001-Warning = {        sql\_managedInstance\_name = "pdsimssqliclear001"        rg\_name = "pd-si-sqlcltr-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqliclear001-Error = {        sql\_managedInstance\_name = "pdsimssqliclear001"        rg\_name = "pd-si-sqlcltr-rg-001"        severity = 1        threshold = 95        #threshold = 1      }         pdsimssqliclearhk002-Warning = {        sql\_managedInstance\_name = "pdsimssqliclearhk002"        rg\_name = "PD-SI-SQLCLTR-RG-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqliclearhk002-Error = {        sql\_managedInstance\_name = "pdsimssqliclearhk002"        rg\_name = "PD-SI-SQLCLTR-RG-001"        severity = 1        threshold = 95        #threshold = 1      }       pdsimssqlil001-Warning = {        sql\_managedInstance\_name = "pdsimssqlil001"        rg\_name = "prd-si-il-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqlil001-Error = {        sql\_managedInstance\_name = "pdsimssqlil001"        rg\_name = "prd-si-il-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      }    } |

**Average\_cpu\_usage.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "DBC-CPU" {    for\_each = var.metric\_alert    name                     = "Alert-AM-Hub-Subscription-SQL-ManagedInstance-${each.key}-AverageCPUPercent"    resource\_group\_name      = var.rg\_name    scopes                   = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/Microsoft.sql/managedinstances/${each.value.sql\_managedInstance\_name}"]    severity                 = each.value.severity    description = "SQL Managed Instance Average CPU Percent Alert"    criteria {      metric\_namespace = "microsoft.sql/managedinstances"      metric\_name      = var.metric\_name      aggregation      = var.time\_aggregation\_method      operator         = var.operator      threshold        = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

**Variable.tf**

|  |
| --- |
| variable "action\_group\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "storage\_space\_used\_mb"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "metric\_alert" {      type = map(object({        sql\_managedInstance\_name = string      rg\_name = string      severity = number      threshold = number      }))      default = {     devcimssqlcts001-Warning = {        sql\_managedInstance\_name = "devcimssqlcts001"        rg\_name = "dev-ci-sql-paas-001"        severity = 2        threshold = 85        #threshold = 2      }        devcimssqlcts001-Error = {        sql\_managedInstance\_name = "devcimssqlcts001"        rg\_name = "dev-ci-sql-paas-001"        severity = 1        threshold = 95        #threshold = 1      }      devcimssqliclearuat001-Warning = {        sql\_managedInstance\_name = "devcimssqliclearuat001"        rg\_name = "dev-ci-sql-paas-001"        severity = 2        threshold = 85        #threshold = 2      }        devcimssqliclearuat001-Error = {        sql\_managedInstance\_name = "devcimssqliclearuat001"        rg\_name = "dev-ci-sql-paas-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqlcts001-Warning = {        sql\_managedInstance\_name = "pdcimssqlcts001"        rg\_name = "pd-ci-sqlcltr-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqlcts001-Error = {        sql\_managedInstance\_name = "pdcimssqlcts001"        rg\_name = "pd-ci-sqlcltr-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqledeposit001-Warning = {        sql\_managedInstance\_name = "pdcimssqledeposit001"        rg\_name = "pd-ci-sqlmi-edeposit-nrg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqledeposit001-Error = {        sql\_managedInstance\_name = "pdcimssqledeposit001"        rg\_name = "pd-ci-sqlmi-edeposit-nrg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqliclear001-Warning = {        sql\_managedInstance\_name = "pdcimssqliclear001"        rg\_name = "pd-ci-sqlcltr-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqliclear001-Error = {        sql\_managedInstance\_name = "pdcimssqliclear001"        rg\_name = "pd-ci-sqlcltr-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqliclearhk002-Warning = {        sql\_managedInstance\_name = "pdcimssqliclearhk002"        rg\_name = "PD-CI-SQLCLTR-RG-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqliclearhk002-Error = {        sql\_managedInstance\_name = "pdcimssqliclearhk002"        rg\_name = "PD-CI-SQLCLTR-RG-001"        severity = 1        threshold = 95        #threshold = 1      }      pdcimssqlil001-Warning = {        sql\_managedInstance\_name = "pdcimssqlil001"        rg\_name = "pd-ci-il-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdcimssqlil001-Error = {        sql\_managedInstance\_name = "pdcimssqlil001"        rg\_name = "pd-ci-il-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdsimssqlcts001-Warning = {        sql\_managedInstance\_name = "pdsimssqlcts001"        rg\_name = "pd-si-sqlcltr-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqlcts001-Error = {        sql\_managedInstance\_name = "pdsimssqlcts001"        rg\_name = "pd-si-sqlcltr-rg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdsimssqledeposit001-Warning = {        sql\_managedInstance\_name = "pdsimssqledeposit001"        rg\_name = "pd-si-sqlmi-edeposit-nrg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqledeposit001-Error = {        sql\_managedInstance\_name = "pdsimssqledeposit001"        rg\_name = "pd-si-sqlmi-edeposit-nrg-001"        severity = 1        threshold = 95        #threshold = 1      }      pdsimssqliclear001-Warning = {        sql\_managedInstance\_name = "pdsimssqliclear001"        rg\_name = "pd-si-sqlcltr-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqliclear001-Error = {        sql\_managedInstance\_name = "pdsimssqliclear001"        rg\_name = "pd-si-sqlcltr-rg-001"        severity = 1        threshold = 95        #threshold = 1      }         pdsimssqliclearhk002-Warning = {        sql\_managedInstance\_name = "pdsimssqliclearhk002"        rg\_name = "PD-SI-SQLCLTR-RG-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqliclearhk002-Error = {        sql\_managedInstance\_name = "pdsimssqliclearhk002"        rg\_name = "PD-SI-SQLCLTR-RG-001"        severity = 1        threshold = 95        #threshold = 1      }       pdsimssqlil001-Warning = {        sql\_managedInstance\_name = "pdsimssqlil001"        rg\_name = "prd-si-il-rg-001"        severity = 2        threshold = 85        #threshold = 2      }        pdsimssqlil001-Error = {        sql\_managedInstance\_name = "pdsimssqlil001"        rg\_name = "prd-si-il-rg-001"        severity = 1        threshold = 95        #threshold = 1      }        }    } |

**Storage\_space\_usage.tf**

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "DBC-stg" {    for\_each = var.metric\_alert    name                     = "Alert-AM-Hub-Subscription-SQL-ManagedInstance-${each.key}-StorageSpaceUsed"    resource\_group\_name      = var.rg\_name    scopes                   = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/Microsoft.sql/managedinstances/${each.value.sql\_managedInstance\_name}"]    severity                 = each.value.severity    description = "SQL Managed Instance Storage Space Used Alert"    criteria {      metric\_namespace = "microsoft.sql/managedinstances"      metric\_name      = var.metric\_name      aggregation      = var.time\_aggregation\_method      operator         = var.operator      threshold        = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* PostGreSqlFlexibleServer –NE: This script is for creating a alert in servers/ databases which include the CPU Percent, storage perecent of the servers/database, oldest transaction, database availability of the servers.
* Availability: Create two file under Availability folder.
* Variable.tf – Create variable file for storing the variable value.
* Availability.tf – Create a resource block for creating a availability metric.

Variable.tf

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_db\_id" {  }  variable "threshold" {    type = number    default = 1  }  variable "operator" {      type = string    default = "LessThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = number    default = 0  }  variable "metric\_name"{      type = string      default = "is\_db\_alive"  }  variable "postgreSQL\_centralIndia" {    type = map(any)    default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Availability.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-Availability-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-DatabaseIsAlive-Critical-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "Database is Alive Alert: Confirms availability"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Cpu Percent: Creating a alert for cpu utilization for database server for metric using cpu percent.
* Create two variable file under the folder : Cpu Percent.
* Variable file: for stroing the value for the variable.
* Cpu\_percent.tf: for creating alert using resource block using the cpu percent metric.

Variable.tf

|  |
| --- |
| variable "action\_group\_db\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "cpu\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Average"  }  variable "severity" {    type = number    default = 2  }  variable "threshold" {    type = number    default = 85  }  variable "postgreSQL\_centralIndia" {    type = map(any)    default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Cpu\_perent.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-cpu\_percent-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-CPUpercent-Warning-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL CPU Percent Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Maximum connection: This script is created for maximum number of connections that is being connected to support a server using maximum connection metric.
* Create a folder under which two file with name space, metric, threshold, time aggregation value, and metric.
* Variable file: To store the variable value.
* Maximum\_connection: Create alert using maximum connection metric.

Variable.tf:

|  |
| --- |
| variable "action\_group\_db\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "max\_connections"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Maximum"  }  variable "severity" {    type = number    default = 2  }  variable "threshold" {    type = number    default = 500  }  variable "postgreSQL\_centralIndia" {    type = map(any)    default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Maximum\_connection.tf

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-max\_connections-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-MaximumConnections-Warning-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL Max Connections Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    window\_size = "PT30M"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Memory Percent: This metric is used to get the memory used in percentage by the databases servers.
* Create two file under a folder named as a memory percent.
* Variable.tf: to store the variable file.
* Memory\_percent: to create resource block for creating memory percent alert.
* Variable.tf

**Variable.tf**

|  |
| --- |
| variable "action\_group\_db\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "memory\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Average"  }  variable "severity" {    type = number    default = 2  }  variable "threshold" {    type = number    default = 85  }  variable "postgreSQL\_centralIndia" {    type = map(any)    default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Memory\_Percent.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-memory\_percent-ci" {    name = "Alert-AM-Prod-Subscription-PostgreSQL-Memorypercent-Warning-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL Memory Percent Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Oldest Transaction: This metric is used to create the active transaction that is being taking long truncation using the longest\_transaction\_time\_sec using the metric, meric namespsace.
* Create two file under the folder which is used to create longest transaction for second.
* Variable.tf: to store the variable value.
* Oldest Transaction.tf: to create resource block for creat the alert for oldest transaction of the server/ databases.

Variable.tf

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_db\_id" {  }  variable "threshold" {    type = number    default = 18000  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Maximum"  }  variable "severity" {    type = number    default = 1  }  variable "metric\_name"{      type = string      default = "longest\_transaction\_time\_sec"  }  variable "postgreSQL\_centralIndia" {    type = map(any)   default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Oldest\_transaction.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-Oldest-Transaction-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-OldestTransaction-Error-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL Oldest Transaction Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Storage Percent: to create the alert for used storage percent of database servers using the metric namespace, metric, threshold.
* Create the two file under the folder for which the storage percent used by utilizing the servers.

**Variable.tf**:

|  |
| --- |
| variable "action\_group\_db\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "storage\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Average"  }  variable "severity" {    type = number    default = 2  }  variable "threshold" {    type = number    default = 85  }  variable "postgreSQL\_centralIndia" {    type = map(any)   default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Storage\_percent.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-storage\_percent-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-Storagepercent-Warning-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL Flexible Server Storage Percent Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* PostGreSqlFlexibleServer –Cental India : This script is for creating a alert in servers/ databases which include the CPU Percent, storage perecent of the servers/database, oldest transaction, database availability of the servers.
* Availability: Create two file under Availability folder.
* Variable.tf – Create variable file for storing the variable value.
* Availability.tf – Create a resource block for creating availability metric.

Variable.tf

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_db\_id" {  }  variable "threshold" {    type = number    default = 1  }  variable "operator" {      type = string    default = "LessThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "severity" {    type = number    default = 0  }  variable "metric\_name"{      type = string      default = "is\_db\_alive"  }  variable "postgreSQL\_centralIndia" {    type = map(any)    default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Availability.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-Availability-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-DatabaseIsAlive-Critical-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "Database is Alive Alert: Confirms availability"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

Cpu Percent: Creating a alert for cpu utilization for database server for metric using cpu percent.

* Create two variable file under the folder : Cpu Percent.
* Variable file: for stroing the value for the variable.
* Cpu\_percent.tf: for creating alert using resource block using the cpu percent metric.

Variable.tf

|  |
| --- |
| variable "action\_group\_db\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "cpu\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Average"  }  variable "severity" {    type = number    default = 2  }  variable "threshold" {    type = number    default = 85  }  variable "postgreSQL\_centralIndia" {    type = map(any)    default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Cpu\_perent.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-cpu\_percent-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-CPUpercent-Warning-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL CPU Percent Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Maximum connection: This script is created for maximum number of connection that is being connected to support a servers using maximum connection metric.
* Create a folder under which two file with name space, metric, threshold, time aggregation value, and metric.
* Variable file: To store the variable value.
* Maximum\_connection: Create alert using maximum connection metric.

**Variable.tf:**

|  |
| --- |
| variable "action\_group\_db\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "max\_connections"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Maximum"  }  variable "severity" {    type = number    default = 2  }  variable "threshold" {    type = number    default = 500  }  variable "postgreSQL\_centralIndia" {    type = map(any)    default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Maximum\_connection.tf

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-max\_connections-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-MaximumConnections-Warning-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL Max Connections Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    window\_size = "PT30M"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Memory Percent: This metric is used to get the memory used in percentage by the databases servers.
* Create two file under a folder named as a memory percent.
* Variable.tf: to store the variable file.
* Memory\_percent: to create resource block for creating memory percent alert.

Variable.tf

|  |
| --- |
| variable "action\_group\_db\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "memory\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Average"  }  variable "severity" {    type = number    default = 2  }  variable "threshold" {    type = number    default = 85  }  variable "postgreSQL\_centralIndia" {    type = map(any)    default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Memory\_Percent.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-memory\_percent-ci" {    name = "Alert-AM-Prod-Subscription-PostgreSQL-Memorypercent-Warning-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL Memory Percent Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Oldest Transaction: This metric is used to create the active transaction that is being taking long truncation using the longest\_transaction\_time\_sec using the metric, meric namespsace.
* Create two file under the folder which is used to create longest transaction for second.
* Variable.tf: to store the variable value.
* Oldest Transaction.tf: to create resource block for creat the alert for oldest transaction of the server/ databases.

Variable.tf

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_db\_id" {  }  variable "threshold" {    type = number    default = 18000  }  variable "operator" {      type = string    default = "GreaterThan"  }  variable "time\_aggregation\_method" {    type = string    default = "Maximum"  }  variable "severity" {    type = number    default = 1  }  variable "metric\_name"{      type = string      default = "longest\_transaction\_time\_sec"  }  variable "postgreSQL\_centralIndia" {    type = map(any)   default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Oldest\_transaction.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-Oldest-Transaction-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-OldestTransaction-Error-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL Oldest Transaction Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Storage Percent: to create the alert for used storage percent of database servers using the metric namespace, metric, threshold.
  + Create the two file under the folder for which the storage percent used by utilizing the servers.

Variable.tf:

|  |
| --- |
| variable "action\_group\_db\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "storage\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Average"  }  variable "severity" {    type = number    default = 2  }  variable "threshold" {    type = number    default = 85  }  variable "postgreSQL\_centralIndia" {    type = map(any)   default = {      postgreSQL = [      {       PostgreSQL\_name = "devcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "devcigispgsql03",       resource\_group\_name = "DEV-CI-GIS-RG-001"      },      {       PostgreSQL\_name = "devciicollectpgsql02",       resource\_group\_name ="dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "preciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "predevcigispgsql02",       resource\_group\_name = "dev-ci-gis-rg-001"      },      {       PostgreSQL\_name = "sitciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      },      {       PostgreSQL\_name = "uatciicollectpgsql02",       resource\_group\_name = "dev-ci-icollect-rg-01"      }      ]    }  } |

Storage\_percent.tf:

|  |
| --- |
| locals {    names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["PostgreSQL\_name"]]    rg\_names = [for sql in var.postgreSQL\_centralIndia.postgreSQL : sql["resource\_group\_name"]]  }  data "azurerm\_postgresql\_flexible\_server" "postgreSQL" {    count = length(local.names)    name = local.names[count.index]    resource\_group\_name = local.rg\_names[count.index]  }  resource "azurerm\_monitor\_metric\_alert" "postgreSQL-storage\_percent-ci" {    name = "Alert-AM-Dev-Subscription-PostgreSQL-Storagepercent-Warning-CentralIndia"    resource\_group\_name = var.rg\_name    scopes = toset(data.azurerm\_postgresql\_flexible\_server.postgreSQL[\*].id)    description = "PostgreSQL Flexible Server Storage Percent Alert"    severity = var.severity    target\_resource\_location = "Central India"    target\_resource\_type = "Microsoft.DBforPostgreSQL/flexibleServers"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/flexibleServers"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_db\_id    }    frequency = "PT5M"  } |

* Data Factory: This metric is used to create failed pipeline of the data factory resources using the metric, metric namespace, threshold, frequency.
* Create a file under the pipeline failed runs.
* Variable.tf: this file is used to create value of variable.
* Pipeline failed run.tf: this file is used to create failed pipeline run in the data factory.

Variable.tf:

|  |
| --- |
| variable "action\_group\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "PipelineFailedRuns"  }  variable "operator" {    default = "GreaterThan"  }  variable "description" {     type    = string    default = "PipelineFailedRuns"  }  variable "time\_aggregation\_method" {    type = string    default = "Count"  }  variable "threshold" {    type = number    default = 0  }  variable "severity" {    type = number    default = 0  }  variable "metric\_alerts" {    type = map(object({      DataFactory\_name = string      rg\_name = string    }))    default = {      devneidmepcdf001 = {        DataFactory\_name = "devneidmepcdf001"        rg\_name = "DEV-NE-EPC-RG-001"        }    }  } |

Failed\_pipeline\_run.tf:

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "ASP\_Memory\_alert" {    for\_each = var.metric\_alerts    name = "Alert-AM-Hub-Subscription-DataFactory-${each.key}-PipelineFailedRuns-Critical"    resource\_group\_name = var.rg\_name    scopes = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/microsoft.datafactory/factories/${each.value.DataFactory\_name}"]    description = var.description    severity = var.severity    criteria {      metric\_namespace = "microsoft.datafactory/factories"      metric\_name = var.metric\_name      aggregation = var.time\_aggregation\_method      operator = var.operator      threshold = var.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    window\_size = "PT15M"    frequency = "PT15M"  } |

* Postgresql single server: This metric is used to create post gre sql single server based on cpu, memory & storage percent using the metric, metric namespace, threshold & frequency.
* Create a folder under which two file.
* Variable.tf: to create a variable file for storing the value of variable
* Cpu percent.tf: to create a resource block for creating a cpu percent metric

Variable.tf:

|  |
| --- |
| variable "action\_group\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "cpu\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    type = string    default = "Average"  }  variable "metric\_alert" {      type = map(object({        PostgreSQL\_name = string      rg\_name = string      severity = number      threshold = number      }))      default = {        devciaquapgsql001-Warning = {        PostgreSQL\_name = "devciaquapgsql001"        rg\_name = "DEV-CI-AQUA-PAAS-001"        severity = 2        threshold = 85        threshold = 2      }      devcidbpgsql001ps-Warning= {        PostgreSQL\_name = "devcidbpgsql001ps"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      devnedbpgsql001ps-Warning= {        PostgreSQL\_name = "devnedbpgsql001ps"        rg\_name = "DEV-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      devnedbpgsql002us-Warning= {        PostgreSQL\_name = "devnedbpgsql002us"        rg\_name = "DEV-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }        pdcigisindbpgsql001-Warning= {        PostgreSQL\_name = "pdcigisindbpgsql001"        rg\_name = "PRD-CI-GIS-RG-001"        severity = 2        threshold = 85      }      pdneidmcredbpgsql001-Warning= {        PostgreSQL\_name = "pdneidmcredbpgsql001"        rg\_name = "PD-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdneilusdbpgsql001-Warning= {        PostgreSQL\_name = "pdneilusdbpgsql001"        rg\_name = "PD-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdneusdbpgsql001-Warning= {        PostgreSQL\_name = "pdneusdbpgsql001"        rg\_name = "PD-NE-USDB-RG-001"        severity = 2        threshold = 85      }     pdsigisindbpgsql001-Warning= {        PostgreSQL\_name = "pdsigisindbpgsql001"        rg\_name = "PRD-CI-GIS-RG-001"        severity = 2        threshold = 85      }          pdweidmcredbpgsql001-Warning= {        PostgreSQL\_name = "pdweidmcredbpgsql001"        rg\_name = "PD-WE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }         pdweilusdbpgsql001-Warning= {        PostgreSQL\_name = "pdweilusdbpgsql001"        rg\_name = "PD-WE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdweusdbpgsql001-Warning= {        PostgreSQL\_name = "pdweusdbpgsql001"        rg\_name = "PD-NE-USDB-RG-001"        severity = 2        threshold = 85      }      }    } |

Cpu\_percent.tf:

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "DBC-CPU" {    for\_each = var.metric\_alert    name                     = "Alert-AM-Hub-Subscription-PostgreSQL-SingleServer-${each.key}-CPUPercent"    resource\_group\_name      = var.rg\_name    scopes                   = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/Microsoft.DBforPostgreSQL/servers/${each.value.PostgreSQL\_name}"]    severity                 = each.value.severity    description = "PostgreSQL CPU Percent Alert"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/servers"      metric\_name      = var.metric\_name      aggregation      = var.time\_aggregation\_method      operator         = var.operator      threshold        = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* Memory Percent: To create a memory percent metric for postgresql single.
* Create a file under the folder for memory percent alert for postgre single server under which how much memory percent used ing single database server.
* variable.tf: to hold the variable value.
* Memory\_percent: to resource block for memory percent.

Variable.tf:

|  |
| --- |
| variable "action\_group\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "memory\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Average"  }  variable "metric\_alert" {      type = map(object({        PostgreSQL\_name = string      rg\_name = string      severity = number      threshold = number      }))      default = {      devciaquapgsql001-Warning = {        PostgreSQL\_name = "devciaquapgsql001"        rg\_name = "DEV-CI-AQUA-PAAS-001"        severity = 2        threshold = 85        threshold = 2      }      devcidbpgsql001ps-Warning= {        PostgreSQL\_name = "devcidbpgsql001ps"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      devnedbpgsql001ps-Warning= {        PostgreSQL\_name = "devnedbpgsql001ps"        rg\_name = "DEV-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      devnedbpgsql002us-Warning= {        PostgreSQL\_name = "devnedbpgsql002us"        rg\_name = "DEV-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }        pdcigisindbpgsql001-Warning= {        PostgreSQL\_name = "pdcigisindbpgsql001"        rg\_name = "PRD-CI-GIS-RG-001"        severity = 2        threshold = 85      }      pdneidmcredbpgsql001-Warning= {        PostgreSQL\_name = "pdneidmcredbpgsql001"        rg\_name = "PD-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdneilusdbpgsql001-Warning= {        PostgreSQL\_name = "pdneilusdbpgsql001"        rg\_name = "PD-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdneusdbpgsql001-Warning= {        PostgreSQL\_name = "pdneusdbpgsql001"        rg\_name = "PD-NE-USDB-RG-001"        severity = 2        threshold = 85      }     pdsigisindbpgsql001-Warning= {        PostgreSQL\_name = "pdsigisindbpgsql001"        rg\_name = "PRD-CI-GIS-RG-001"        severity = 2        threshold = 85      }          pdweidmcredbpgsql001-Warning= {        PostgreSQL\_name = "pdweidmcredbpgsql001"        rg\_name = "PD-WE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }         pdweilusdbpgsql001-Warning= {        PostgreSQL\_name = "pdweilusdbpgsql001"        rg\_name = "PD-WE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdweusdbpgsql001-Warning= {        PostgreSQL\_name = "pdweusdbpgsql001"        rg\_name = "PD-NE-USDB-RG-001"        severity = 2        threshold = 85      }      }    } |

memory\_percent.tf:

|  |
| --- |
| # data "azurerm\_postgresql\_flexible\_server" "example" {  #   for\_each = var.metric\_alert  #   name                = each.value.PostgreSQL\_name  #   resource\_group\_name = each.value.rg\_name  # }  resource "azurerm\_monitor\_metric\_alert" "DBC-Memory" {    for\_each = var.metric\_alert    name                     = "Alert-AM-Hub-Subscription-PostgreSQL-SingleServer-${each.key}-MemoryPercent"    resource\_group\_name      = var.rg\_name    scopes                   = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/Microsoft.DBforPostgreSQL/servers/${each.value.PostgreSQL\_name}"]    severity                 = each.value.severity    description = "PostgreSQL Memory Percent Alert"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/servers"      metric\_name      = var.metric\_name      aggregation      = var.time\_aggregation\_method      operator         = var.operator      threshold        = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* Storage Percent: to create the storage percent alert for database servers using the metric, metric namespace, threshold, frequency.
* Create a file under the folder for storage percent.

Variable.tf:

|  |
| --- |
| variable "action\_group\_id" {    description = "Action Group Id"  }  variable "rg\_name" {    }  variable "metric\_name" {    default = "storage\_percent"  }  variable "operator" {    default = "GreaterThanOrEqual"  }  variable "time\_aggregation\_method" {    default = "Average"  }  variable "metric\_alert" {      type = map(object({        PostgreSQL\_name = string      rg\_name = string      severity = number      threshold = number      }))      default = {      devciaquapgsql001-Warning = {        PostgreSQL\_name = "devciaquapgsql001"        rg\_name = "DEV-CI-AQUA-PAAS-001"        severity = 2        threshold = 85        threshold = 2      }      devcidbpgsql001ps-Warning= {        PostgreSQL\_name = "devcidbpgsql001ps"        rg\_name = "DEV-CI-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      devnedbpgsql001ps-Warning= {        PostgreSQL\_name = "devnedbpgsql001ps"        rg\_name = "DEV-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      devnedbpgsql002us-Warning= {        PostgreSQL\_name = "devnedbpgsql002us"        rg\_name = "DEV-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }        pdcigisindbpgsql001-Warning= {        PostgreSQL\_name = "pdcigisindbpgsql001"        rg\_name = "PRD-CI-GIS-RG-001"        severity = 2        threshold = 85      }      pdneidmcredbpgsql001-Warning= {        PostgreSQL\_name = "pdneidmcredbpgsql001"        rg\_name = "PD-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdneilusdbpgsql001-Warning= {        PostgreSQL\_name = "pdneilusdbpgsql001"        rg\_name = "PD-NE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdneusdbpgsql001-Warning= {        PostgreSQL\_name = "pdneusdbpgsql001"        rg\_name = "PD-NE-USDB-RG-001"        severity = 2        threshold = 85      }      pdsigisindbpgsql001-Warning= {        PostgreSQL\_name = "pdsigisindbpgsql001"        rg\_name = "PRD-CI-GIS-RG-001"        severity = 2        threshold = 85      }        pdweidmcredbpgsql001-Warning= {        PostgreSQL\_name = "pdweidmcredbpgsql001"        rg\_name = "PD-WE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }        pdweilusdbpgsql001-Warning= {        PostgreSQL\_name = "pdweilusdbpgsql001"        rg\_name = "PD-WE-POSTGRE-PAAS-001"        severity = 2        threshold = 85      }      pdweusdbpgsql001-Warning= {        PostgreSQL\_name = "pdweusdbpgsql001"        rg\_name = "PD-NE-USDB-RG-001"        severity = 2        threshold = 85      }      }    } |

Storage\_percent.tf:

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "DBC-storage" {    for\_each = var.metric\_alert    name                     = "Alert-AM-Hub-Subscription-PostgreSQL-SingleServer-${each.key}-StoragePercent"    resource\_group\_name      = var.rg\_name    scopes                   = ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/Microsoft.DBforPostgreSQL/servers/${each.value.PostgreSQL\_name}"]    severity                 = each.value.severity    description = "PostgreSQL Storage Percent Alert"    criteria {      metric\_namespace = "Microsoft.DBforPostgreSQL/servers"      metric\_name      = var.metric\_name      aggregation      = var.time\_aggregation\_method      operator         = var.operator      threshold        = each.value.threshold    }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* AKS - Cpu Usage Percentage: This metric in AKS will trigger an alert when the cpu usage percentage of all kubernetes services exceeds its threshold value.
* Create a folder in which two files will create.
* Variable.tf : To store the vale of the variable
* Cpu\_usage\_percentage.tf: to create the alert using resource block.

Variable.tf:

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Aggregated  average CPU utilization  measured in percentage across the cluster."  }  variable "metric\_name" {    type = string    default = "node\_cpu\_usage\_percentage"  }  variable "operator" {    type = string    default = "GreaterThanOrEqual"  }  variable "aggregation" {    type = string    default = "Average"  }    variable "metric\_alert" {    type = map(object({      kubernetes\_name = string      rg\_name = string      severity = number      threshold = number      }))    default = {        devciakscluster01-Warning= {       kubernetes\_name= "devciakscluster01"       rg\_name = "DEV-CI-AKS-RG-001"       severity = 2       threshold = 80        },      devciakscluster01-Error= {       kubernetes\_name= "devciakscluster01"       rg\_name = "DEV-CI-AKS-RG-001"       severity = 1       threshold = 95        },      devneakscluster01-Warning= {       kubernetes\_name= "devneakscluster01"       rg\_name = "DEV-NE-AKS-RG-001"       severity = 2       threshold = 80        },      devneakscluster01-Error= {       kubernetes\_name= "devneakscluster01"       rg\_name = "DEV-NE-AKS-RG-001"       severity = 1       threshold = 95        },      pdciakscluster01-Warning= {       kubernetes\_name= "pdciakscluster01"       rg\_name = "PD-CI-AKS-RG-001"       severity = 2       threshold = 80        },     pdciakscluster01-Error= {       kubernetes\_name= "pdciakscluster01"       rg\_name = "PD-CI-AKS-RG-001"       severity = 1       threshold = 95        },      pdneakscluster01-Warning= {       kubernetes\_name= "pdneakscluster01"       rg\_name = "PD-NE-AKS-RG-001"       severity = 2       threshold = 80        },      pdneakscluster01-Error= {       kubernetes\_name= "pdneakscluster01"       rg\_name = "PD-NE-AKS-RG-001"       severity = 1       threshold = 95        },      pdsiakscluster01-Warning= {       kubernetes\_name= "pdsiakscluster01"       rg\_name = "pd-si-aks-rg-001"       severity = 2       threshold = 80        },      pdsiakscluster01-Error= {       kubernetes\_name= "pdsiakscluster01"       rg\_name = "pd-si-aks-rg-001"       severity = 1       threshold = 95        },      pdweakscluster01-Warning= {       kubernetes\_name= "pdweakscluster01"       rg\_name = "PD-WE-AKS-RG-001"       severity = 2       threshold = 80        },       pdweakscluster01-Error= {       kubernetes\_name= "pdweakscluster01"       rg\_name = "PD-WE-AKS-RG-001"       severity = 1       threshold = 95        }      }  } |

Cpu\_usage\_percentage.tf:

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "CPUPercentage" {    for\_each = var.metric\_alert    name                = "Alert-AM-Hub-Subscription-AKS-CPU-Percentage-${each.key}"    resource\_group\_name =  var.rg\_name    scopes =            ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/Microsoft.ContainerService/managedClusters/${each.value.kubernetes\_name}"]    description         = var.description    severity            = each.value.severity        criteria {      metric\_namespace  = "Microsoft.ContainerService/ManagedClusters"      metric\_name       = var.metric\_name      operator          = var.operator      threshold         = each.value.threshold      aggregation       = var.aggregation      }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* Azure Cluster services:
* DiskUsage Percentage: This metric in AKS will trigger an alert when the node disk usage percentage of all kubernetes services exceeds its threshold value.
* Create a folder in which two files will create.
* Variable.tf : To store the vale of the variable
* disk\_usage\_percentage.tf: to create the alert using resource block using the node disk usage percentage.

Variable.tf:

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Disk percentge usage in device."  }  variable "metric\_name" {    type = string    default = "node\_disk\_usage\_percentage"  }  variable "operator" {    type = string    default = "GreaterThanOrEqual"  }  variable "aggregation" {    type = string    default = "Average"  }    variable "metric\_alert" {    type = map(object({      kubernetes\_name = string      rg\_name = string      severity = number      threshold = number      }))      default = {        devciakscluster01-Warning= {       kubernetes\_name= "devciakscluster01"       rg\_name = "DEV-CI-AKS-RG-001"       severity = 2       threshold = 80        },      devciakscluster01-Error= {       kubernetes\_name= "devciakscluster01"       rg\_name = "DEV-CI-AKS-RG-001"       severity = 1       threshold = 95        },      devneakscluster01-Warning= {       kubernetes\_name= "devneakscluster01"       rg\_name = "DEV-NE-AKS-RG-001"       severity = 2       threshold = 80        },      devneakscluster01-Error= {       kubernetes\_name= "devneakscluster01"       rg\_name = "DEV-NE-AKS-RG-001"       severity = 1       threshold = 95        },      pdciakscluster01-Warning= {       kubernetes\_name= "pdciakscluster01"       rg\_name = "PD-CI-AKS-RG-001"       severity = 2       threshold = 80        },     pdciakscluster01-Error= {       kubernetes\_name= "pdciakscluster01"       rg\_name = "PD-CI-AKS-RG-001"       severity = 1       threshold = 95        },      pdneakscluster01-Warning= {       kubernetes\_name= "pdneakscluster01"       rg\_name = "PD-NE-AKS-RG-001"       severity = 2       threshold = 80        },      pdneakscluster01-Error= {       kubernetes\_name= "pdneakscluster01"       rg\_name = "PD-NE-AKS-RG-001"       severity = 1       threshold = 95        },      pdsiakscluster01-Warning= {       kubernetes\_name= "pdsiakscluster01"       rg\_name = "pd-si-aks-rg-001"       severity = 2       threshold = 80        },      pdsiakscluster01-Error= {       kubernetes\_name= "pdsiakscluster01"       rg\_name = "pd-si-aks-rg-001"       severity = 1       threshold = 95        },      pdweakscluster01-Warning= {       kubernetes\_name= "pdweakscluster01"       rg\_name = "PD-WE-AKS-RG-001"       severity = 2       threshold = 80        },       pdweakscluster01-Error= {       kubernetes\_name= "pdweakscluster01"       rg\_name = "PD-WE-AKS-RG-001"       severity = 1       threshold = 95        }      }  } |

disk\_usage\_percentage.tf:

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "Disk\_Percentage" {    for\_each = var.metric\_alert    name                = "Alert-AM-Hub-Subscription-AKS-Disk-Percentage-${each.key}"    resource\_group\_name =  var.rg\_name    scopes =            ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/Microsoft.ContainerService/managedClusters/${each.value.kubernetes\_name}"]    description         = var.description    severity            = each.value.severity        criteria {      metric\_namespace  = "Microsoft.ContainerService/ManagedClusters"      metric\_name       = var.metric\_name      operator          = var.operator      threshold         = each.value.threshold      aggregation       = var.aggregation      }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

* Memory Working Set Percentage: This metric in AKS will trigger an alert when the node memeory working set usage of all kubernetes services exceeds its threshold value.
* Create a folder in which two files will create.
* Variable.tf : To store the vale of the variable
* Memory\_working\_set\_percentage.tf: to create the alert using resource block using the node disk usage percentage

Variable.tf:

|  |
| --- |
| variable "rg\_name" {    type        = string    description = "Name of the Resource group in which to deploy shared resources"  }  variable "action\_group\_id" {  }  variable "description" {      type = string      default = "Container working set memory used in percent."  }  variable "metric\_name" {    type = string    default = "node\_memory\_working\_set\_percentage"  }  variable "operator" {    type = string    default = "GreaterThanOrEqual"  }  variable "aggregation" {    type = string    default = "Average"  }  variable "metric\_alert" {    type = map(object({      kubernetes\_name = string      rg\_name = string      severity = number      threshold = number    }))      default = {        devciakscluster01-Warning= {       kubernetes\_name= "devciakscluster01"       rg\_name = "DEV-CI-AKS-RG-001"       severity = 2       threshold = 80        },      devciakscluster01-Error= {       kubernetes\_name= "devciakscluster01"       rg\_name = "DEV-CI-AKS-RG-001"       severity = 1       threshold = 95        },      devneakscluster01-Warning= {       kubernetes\_name= "devneakscluster01"       rg\_name = "DEV-NE-AKS-RG-001"       severity = 2       threshold = 80        },      devneakscluster01-Error= {       kubernetes\_name= "devneakscluster01"       rg\_name = "DEV-NE-AKS-RG-001"       severity = 1       threshold = 95        },      pdciakscluster01-Warning= {       kubernetes\_name= "pdciakscluster01"       rg\_name = "PD-CI-AKS-RG-001"       severity = 2       threshold = 80        },     pdciakscluster01-Error= {       kubernetes\_name= "pdciakscluster01"       rg\_name = "PD-CI-AKS-RG-001"       severity = 1       threshold = 95        },      pdneakscluster01-Warning= {       kubernetes\_name= "pdneakscluster01"       rg\_name = "PD-NE-AKS-RG-001"       severity = 2       threshold = 80        },      pdneakscluster01-Error= {       kubernetes\_name= "pdneakscluster01"       rg\_name = "PD-NE-AKS-RG-001"       severity = 1       threshold = 95        },      pdsiakscluster01-Warning= {       kubernetes\_name= "pdsiakscluster01"       rg\_name = "pd-si-aks-rg-001"       severity = 2       threshold = 80        },      pdsiakscluster01-Error= {       kubernetes\_name= "pdsiakscluster01"       rg\_name = "pd-si-aks-rg-001"       severity = 1       threshold = 95        },      pdweakscluster01-Warning= {       kubernetes\_name= "pdweakscluster01"       rg\_name = "PD-WE-AKS-RG-001"       severity = 2       threshold = 80        },       pdweakscluster01-Error= {       kubernetes\_name= "pdweakscluster01"       rg\_name = "PD-WE-AKS-RG-001"       severity = 1       threshold = 95        }      }  } |

Memory\_working\_set\_percentage.tf:

|  |
| --- |
| resource "azurerm\_monitor\_metric\_alert" "memory\_Working\_set\_percentage" {    for\_each = var.metric\_alert    name                = "Alert-AM-Hub-Subscription-AKS-Memory-Workingset-Percentage-${each.key}"    resource\_group\_name =  var.rg\_name    scopes =            ["/subscriptions/fe8607f3-f657-4c44-af82-f50b0693554a/resourceGroups/${each.value.rg\_name}/providers/Microsoft.ContainerService/managedClusters/${each.value.kubernetes\_name}"]    description         = var.description    severity            = each.value.severity      criteria {      metric\_namespace  = "Microsoft.ContainerService/ManagedClusters"      metric\_name       = var.metric\_name      operator          = var.operator      threshold         = each.value.threshold      aggregation       = var.aggregation      }    action {      action\_group\_id = var.action\_group\_id    }    frequency = "PT5M"  } |

# v5. ALERTS TO BE CONFIGURED WITH TERRAFORM CODES IN AZURE DEVOPS.

[**DBCHEQUES\_Monitor\_Alert\_Parameters**](https://hclo365.sharepoint.com/:x:/r/sites/EOPS-AUTONOMICS/_layouts/15/Doc2.aspx?action=edit&sourcedoc=%7Be6c1fb20-8029-4419-b5d6-4acfc2ce9197%7D&wdOrigin=TEAMS-ELECTRON.teamsSdk_ns.bim&wdExp=TEAMS-CONTROL&wdhostclicktime=1700727386895&web=1)

## Platform Services, Virtual Machines & OS Metrices

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Technology** | **Metric Display Name** | **Metric Name** | **Unit** | **Description** | **Frequency ( Minutes ) (Minutes)** | **Thresholds** | | **Number of violations** | **Evalution Period** |
| **Warning** | **Critical** |  |  |
| **Windows OS** | Total CPU Utilization Percentage | Processor(\_Total)\% Processor Time | Percent | Total CPU Utilization Percentage | **5** | **85** | **90** | **4** | **20 Mints** |
| Logical Disk Free Space | Logical Disk(\*)\% Free Space | Percent | Logical Disk Free Space | **5** | **15** | **10** | **4** | **20 Mints** |
| Memory utilization | Memory\Available Bytes | Bytes | Memory Used percentage | **5** | **85** | **90** | **4** | **20 Mints** |
| VM Availability Matrix | VM Availability Matrix | Count | The metric allows you to track the pulse of your VMs—during expected behavior, the metric displays a value of 1. In response to any VM availability disruptions, the metric dips to a 0 for the duration of impact. | **5** |  | **>1** |  |  |
| Data Disk IOPS Consumed Percentage | Data Disk IOPS Consumed Percentage | Percent | Percentage of data disk I/Os consumed per minute | **5** | **85** | **90** | **4** | **20 Mints** |
| System Event Log | Windows Shutdown/Start |  | 41, 1074, 6006, 6008 / 6005 | **5** |  | **Critical** |  |  |
| **LINUX OS** | Total CPU Utilization Percentage | Processor(\_Total)\% Processor Time | Percent | Total CPU Utilization Percentage | **5** | **85** | **90** | **4** | **20 Mints** |
| Logical Disk UsedSpace | Logical Disk(\*)\% Used Space | Percent | Logical Disk UsedSpace | **5** | **85** | **90** | **4** | **20 Mints** |
| Memory Used Swap Space | Memory\% Used Swap Space | Percent | Memory Used Swap Space | **5** | **50** | **80** | **4** | **20 Mints** |
| Memory Used percentage | Memory \% Used Memory | Percent | Memory Used percentage | **5** | **85** | **90** | **4** | **20 Mints** |
| **Backup/Recovery** | BackupHealthEvent | Backup Health Events - failure (preview) | Count | The count of health events pertaining to backup job health | **15** |  | **1** | **1** | **15 Mint** |
| **Load Balancer** | DipAvailability | Health Probe Status | Count | Average Load Balancer health probe status per time duration | **5** | **1** | **2** |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

## SQL Database Metrics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cloud Scope** | **DB Type** | **Service Name** | **Metrics** | **Thresholds** | |
| Azure | PaaS / Managed Instance | MSSQL | CPU Percentage | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MSSQL | DTU Percentage | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MSSQL | storage\_percent/ Data space used percent | >=85% | >= 90% |
| Azure | PaaS / Managed Instance | MSSQL | Failed Connections | If there are more that 5 consecutive failed connection for same user. | If there are more that 10 consecutive failed connection for same user. |
| Azure | PaaS / Managed Instance | MSSQL | SQL instance CPU percent | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MSSQL | SQL instance memory percent | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MSSQL | CPU Limit | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MSSQL | CPU Used | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MSSQL | Data IO percentage | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MSSQL | Log IO | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MSSQL | Deadlocks | 1 | 3 |
| Azure | PaaS / Managed Instance | MSSQL | Sessions count | >=25 | >=30 |

## PAAS Services

### App Service Plan Metrics

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Technology** | **Metric Display Name** | **Metric Name** | **Unit** | **Description** | **Frequency ( Minutes )** **(Minutes)** | **Thresholds** | |
| **Warning** | **Critical** |
| **App Service Plan** | CpuPercentage | CPU Percentage | Percent | CpuPercentage | 15 | 80 | 90 |
| MemoryPercentage | Memory Percentage | Percent | MemoryPercentage | 15 | 80 | 90 |

### Application Insights Metrics

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Technology** | **Metric Display Name** | **Metric Name** | **Unit** | **Description** | **Frequency ( Minutes )** **(Minutes)** | **Thresholds** | |
| **Warning** | **Critical** |
| **App Insight** | Availability | Availability | Percent | Availability | 15 | 95 | 85 |
| Process CPU | Process CPU |  | Process CPU | 15 | 80 | 90 |
| Browser exceptions | Browser exceptions | Count | Browser exceptions | 30 | 400 | 600 |
| Browser page load time | Browser page load time | MilliSeconds | Browser page load time | 15 | 3000 | 5000 |
| dependency duration | Client processing time | MilliSeconds | Client processing time | 15 | 6000 | 8000 |
| Failed requests | Failed requests | Count | Failed requests | 15 | 200 | 250 |
| Server response time | Server response time | MilliSeconds | Server response time | 15 | 6000 | 7500 |
| Server exceptions | Server exceptions | Count | Server exceptions | 15 | 250 | 400 |

### Application Gateway Metrics

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Technology | Metric Display Name | Metric Name | Unit | Frequency ( Minutes ) | Thresholds (warning/Critical) | |
| Application Gateway | Backend Health probe | Backend Health probe | Count | 5 | 1 | 2 |
|  |  |  |  |  |  |

### Traffic Manager

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Technology | Metric Display Name | Metric Name | Unit | Frequency ( Minutes ) | Thresholds (warning/Critical) | |
| (Minutes) | Warning | Critical |
| Traffic Manager profile | Backend Health probe | Backend Health probe | NA | 5 | 1 | 2 |

### Function App

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Technology | Metric Display Name | Metric Name | Unit | Description | Frequency ( Minutes ) | Thresholds (warning/Critical) | |
| Function App | Http Server Errors | Http5xx | Count | The count of requests resulting in an HTTP status code = 500 but < 600. | 15 | 50 | 100 |
|  |  |  |  |  |  |  |
| Response Time | HttpResponseTime | Seconds | The time taken for the app to serve requests, in seconds. For WebApps and FunctionApps. | 15 | 10 | 15 |

### Azure Logic Apps

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Technology | Metric Display Name | Metric Name | Unit | Frequency ( Minutes ) | Thresholds (warning/Critical) | |
| (Minutes) | Warning | Critical |
| Logic App | Actions Failed | Actions Failed | count | 5 | NA | > 0 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

### MySQL Database Metrics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cloud Scope** | **DB Type** | **Service Name** | **Metrics** | **Thresholds** | |
| Azure | PaaS / Managed Instance | MySQL | CPU Percentage | >=85% | >=90% |
| Azure | PaaS / Managed Instance | MySQL | storage\_percent | >=85% | >= 90% |

### Postgre SQL DB

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cloud Scope** | **DB Type** | **Service Name** | **Metrics** | **Thresholds/ Frequency** | |
| Azure | PostGreSQl Flexible Server | PostGreSql | CPU Percentage | >=85%/ 5 min |  |
| Azure | PostGreSQl Flexible Server | PostGreSql | storage\_percent | >=85%/5 min |  |
| Azure | PostGreSQl Flexible Server | PostGreSql | memory\_percent | >=85%/5 min |  |
| Azure | PostGreSQl Flexible Server | PostGreSql | Database is Alive | yes/no/5 min |  |
| Azure | PostGreSQl Flexible Server | PostGreSql | Oldest Transaction | 18000(Seconds)/ 5 min |  |
| Azure | PostGreSQl Flexible Server | PostGreSql | Max Connections | >= 500/5 min |  |

### Azure Data factory

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Technology | Metric Display Name | Metric Name | Unit | Frequency ( Minutes ) | Thresholds (warning/Critical) | |
| (Minutes) | Warning | Critical |
| Data Factory | PipelineFailedRuns | Failed pipeline runs metrics | Count | 5 |  | >0 |

### Azure Kubernetes Services

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Technology | Metric Display Name | Metric Name | Description | Frequency ( Minutes ) | Thresholds (warning/Critical) | |
| (Minutes) | Warning | Critical |
| AKS Cluster | CPU Usage Percentage | CPU Usage Percentage | CPU Usage Percentage | 5 | 80/2 | 95/1 |
|  |  |  |  |  |  |  |
| AKS Cluster | Disk Usage Percentage | Disk Usage Percentage | Disk Usage Percentage | 5 | 80/2 | 95/1 |
| AKS Cluster | Memory Working Set Percentage | Memory Working Set Percentage | Memory Working Set Percentage | 5 | 80/2 | 95/1 |
| AKS Cluster | Cluster Health | Cluster Health | Cluster Health | 5 | >1 |  |
|  |  |  |  |  |  |  |

# TERRAFORM CODE DETAILS FOR ALL ALERTS - LOCATION

[DBCheque-AM - Repos (azure.com)](https://dev.azure.com/DBCHEQUES/DBCHEQUES/_git/DBCheque-AM)

# CODE DEPLOYMENT STEPS VIA AZURE DEVOPS SYSTEM

Sequential Step by Steps

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Steps for DBCHEQUES** | **Status** |  |
| 1 | Create Service Connection | Done |  |
| 2 | Install Terraform Extension from the marketplace | Done |  |
| 3 | Push the code to the Azure repos | Done |  |
| 4 | Code Explanation   * **Enable System Identity** & Install AMA Agent using PS Code * Create Resource Group to host all AM resources * Create Storage Account & Container for storing TF State file(make sure to lock state file) * Create Log Analytics Workspace * Create Action Group & add action to send notification through email/sms. * Create Data Collection Rule and adding destination workspace (LAW) and target resources (VM) * Create Alerts | Done |  |
| 5 | Create Release Pipeline   * Add Artifacts and Tasks * Create Release and deploy the code | Done |  |
|  |  |  |  |

# BEST PRACTICES

## Log Analytics Workspaces Best Practice

| **LAW Best Practice** |
| --- |
| **There are several ways to optimize the cost of Azure Log Analytics, including:** |
| * Log less data by reducing the log level or using a **daily cap to limit the ingestion volume.** |
| * Use **shorter data retention times** in the Log Analytics workspace. |
| * **Offload logs to cheaper storage** such as an Azure Storage account (Archive Storage). |
| * Use the **commitment tier pricing model** to get discounts based on your expected usage. |
| * **Use fewer workspaces** to avoid duplication and complexity. |
|  |
| **How to optimize Azure Log Analytics Workspace Cost** |
| * Azure Monitor and log analytics pricing methodology: |
| * Data Ingestion |
| * Data Retention period |
| * Data export |
|  |
| * To optimize and do better in log analytics cost management we need to optimize all these **three activities**mentioned above. |
| * Capture only the logs which are important for your monitoring purpose. If you are using web app service we **do not need platform logs**and azure resource level logs which we can turn off. |
| * We need to choose the **cheapest region** to create and store log analytics workspace. |
| * **Purge the logs** which are quite old and not needed in future. |
| * If we have very high volume of the log **ingestion**then it would be prudent to opt for Azure **commitment tier. FYI, commitment only works for 100 TB and more.** |
| * In case we need to export the log analytics data, rather than exporting all the data, we can filter it and send only relevant log data. |

## Data Collection Rules

| **Category** | **Best practice** | **Explanation** | **Impact area** |
| --- | --- | --- | --- |
| **Data Collection** | Define the observability scope | Defining the observability scope is key to an easier and successful DCR management and organization observability scope. It will help clarifying what the collection need is, and from which target virtual machine it should be performed. As previously explained, an observability scope could be a set of virtual machines running software that is common to a specific application, a set of common information for the IT department, etc. As an example, collecting the basic operating system performance counters, such as CPU utilization, available memory, and free disk space, could be seen as a scope for your Central IT Management. | Not having a clearly defined scope does not bring clarity and does not allow for a proper management. |
| **Data Collection** | Create DCRs specific to the observability scope | Creating separate DCRs based on the observability scope is key for easy maintenance. It will allow you to easily associate the DCRs to the relevant target virtual machines. | Why creating a single DCR that collects operating system performance counters plus web server counters and database counters all together? This approach, not only will force each associated virtual machine to transfer, process and execute configuration that is outside of the scope but will also require more effort when the DCR configuration needs to be updated. Think about managing a template that includes unnecessary entries; this situation is less than ideal and leaves room for errors. |
| **Data Collection** | Create DCR specific to data source type inside the defined observability scope(s) | Creating separate DCRs for performance and events will help in both managing the configuration and the association with granularity based on the target machines. For instance, creating a DCR to collect both events and performance counters could result in a suboptimal approach. There could be situations in which a given machine (or set of machines) does not have the event logs or performance counters configured in the DCR. In this situation, the virtual machine(s) will be forced to process and execute a configuration that is not necessary according to the software installed on it. | [Not using different DCRs will force each associated virtual machine to transfer process and execute configuration that might be not applicable according to the installed software. An excessive compute resource consumption and errors in processing configuration might happen causing the Azure Monitor Agent (AMA) becoming unresponsive. Moreover, collecting unnecessary data will increase data ingestion costs.](https://learn.microsoft.com/en-us/azure/azure-monitor/overview) |
| **Data Destination** | Create different DCR based on the destination | DCRs have the capability of sending data to multiple different destinations, like Azure Monitor Metrics and Azure Monitor Logs, simultaneously. Having DCR(s) specific to destination is helpful in managing the data sovereign or law requirements. Since, being compliant might require to send data only to allowed repositories created in allowed regions, having different DCRs allows for a better granular destination targeting | Not separating DCRs based on the data destination, might result in being not compliant with data handling, privacy and access requirements and could make unnecessary data collection resulting in unexpected costs. |

## VM Monitoring Best Practice

| **Reliability** | |
| --- | --- |
| **Design Checklist** | **Configuration** |
| **Create availability alert rules for Azure VMs.** | Use the availability metric (preview) to track when an Azure VM is running. While you can quickly enable an availability alert rule for an individual machine using recommended alerts, a single alert rule targeting a resource group or subscription enables availability alerting for all VMs in that scope for a particular region. This is easier to manage than creating an alert rule for each VM and ensures that any new VMs created in the scope are automatically monitored. This alert rule does not require the Azure Monitor agent to be installed on the VM, but it is not available for VMs outside of Azure. |
| **Create agent heartbeat alert rule to verify agent health.** | The Azure Monitor agent sends a heartbeat to the Log Analytics workspace every minute. Use a log query alert rule using the agent heartbeat to be alerted when an agent stops sending heartbeats, which is an indicator that either the VM is down or the agent is unhealthy and client workloads are not being monitored. This alert rule requires that the Azure Monitor agent is installed on the VM and applies to both Azure and non-Azure VMs. |

| **Security** | |
| --- | --- |
| **Design Checklist** | **Configuration** |
| **Use other services for security monitoring of your VMs.** | While Azure Monitor can collect security events from your VMs, it isn't intended to be used for security monitoring. Azure includes multiple services such as Microsoft Defender for Cloud and Microsoft Sentinel that together provide a complete security monitoring solution. See Security monitoring for a comparison of these services. |
| **Consider using Azure private link for VMs to connect to Azure Monitor using a private endpoint.** | Connections to public endpoints are secured with end-to-end encryption. If you require a private endpoint, you can use Azure private link to allow your VMs to connect to Azure Monitor through authorized private networks. Private link can also be used to force workspace data ingestion through ExpressRoute or a VPN. See Design your Azure Private Link setup to determine the best network and DNS topology for your environment. |

| **Cost optimization** | |
| --- | --- |
| **Design Checklist** | **Configuration** |
| **Migrate from Log Analytics agent to Azure Monitor agent for granular data filtering.** | If you still have VMs with the Log Analytics agent, migrate them to Azure Monitor agent so you can take advantage of better data filtering and use unique configurations with different sets of VMs. Configuration for data collection by the Log Analytics agent is done on the workspace, so all agents receive the same configuration. Data collection rules used by Azure Monitor agent can be tuned to the specific monitoring requirements of different sets of VMs. The Azure Monitor agent also allows you to use transformations to filter data being collected. |
| **Filter data that you don't require from agents.** | Reduce your data ingestion costs by filtering data that you don't use for alerting or analysis. See Monitor virtual machines with Azure Monitor: Collect data for guidance on data to collect for different monitoring scenarios and Control costs for specific guidance on filtering data to reduce your costs. |
| **Determine whether you'll use VM insights and what data to collect.** | VM insights is a great feature to quickly get started with monitoring your VMs and provides powerful features such as Map and performance trend views. If you don't use the Map feature or the data that it collects, then you should disable collection of processes and dependency data in your VM insights configuration to save on data ingestion costs. |
| **Reduce polling frequency of performance counters.** | If you're using a data collection rule to send performance data to your Log Analytics workspace, you can reduce their polling frequency to reduce the amount of data collected. |
| **Ensure that VMs aren't sending duplicate data.** | If you multi-home agents or you create similar data collection rules, make sure you are sending unique data to each workspace. See Analyze usage in Log Analytics workspace for guidance on analyzing your collected data to make sure you are not collecting duplicate data. If you're migrating between agents, continue to use the Log Analytics agent until you migrate to the Azure Monitor agent rather than using both together unless you can ensure that each is collecting unique data. |

| **Operational Excellence** | |
| --- | --- |
| **Design Checklist** | **Configuration** |
| **Use Azure Policy to deploy agents and assign data collection rules.** | Azure Policy allows you to have agents automatically deployed to sets of existing VMs and any new VMs that are created. This ensures that all VMs are monitored with minimal intervention by administrators. If you use VM insights, see Enable VM insights by using Azure Policy. If you want to manage Azure Monitor agent without VM insights, see Enable Azure Monitor Agent by using Azure Policy. See Resource Manager template samples for data collection rules in Azure Monitor for a template to create a data collection rule association. |
| **Establish a strategy for structure of data collection rules.** | Data collection rules define data to collect from virtual machines with the Azure Monitor agent and where to send that data. Each DCR can include multiple collection scenarios and be associated with any number of VMs. Establish a strategy for configuring DCRs to collect only required data for different groups of VMs while minimizing the number of DCRs that you need to manage. |

# IMPORTANT LINKS

* [azurerm\_monitor\_metric\_alert | Resources | hashicorp/azurerm | Terraform | Terraform Registry](https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/monitor_metric_alert)

This type of alert is used to setup monitoring for predefined metrics for various azure services such as application insights, web app /function app etc.

* [azurerm\_monitor\_scheduled\_query\_rules\_alert\_v2 | Resources | hashicorp/azurerm | Terraform | Terraform Registry](https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/monitor_scheduled_query_rules_alert_v2)

This alert is primarily used for monitoring the guest metrics of Windows/Linux operating systems using KQL.

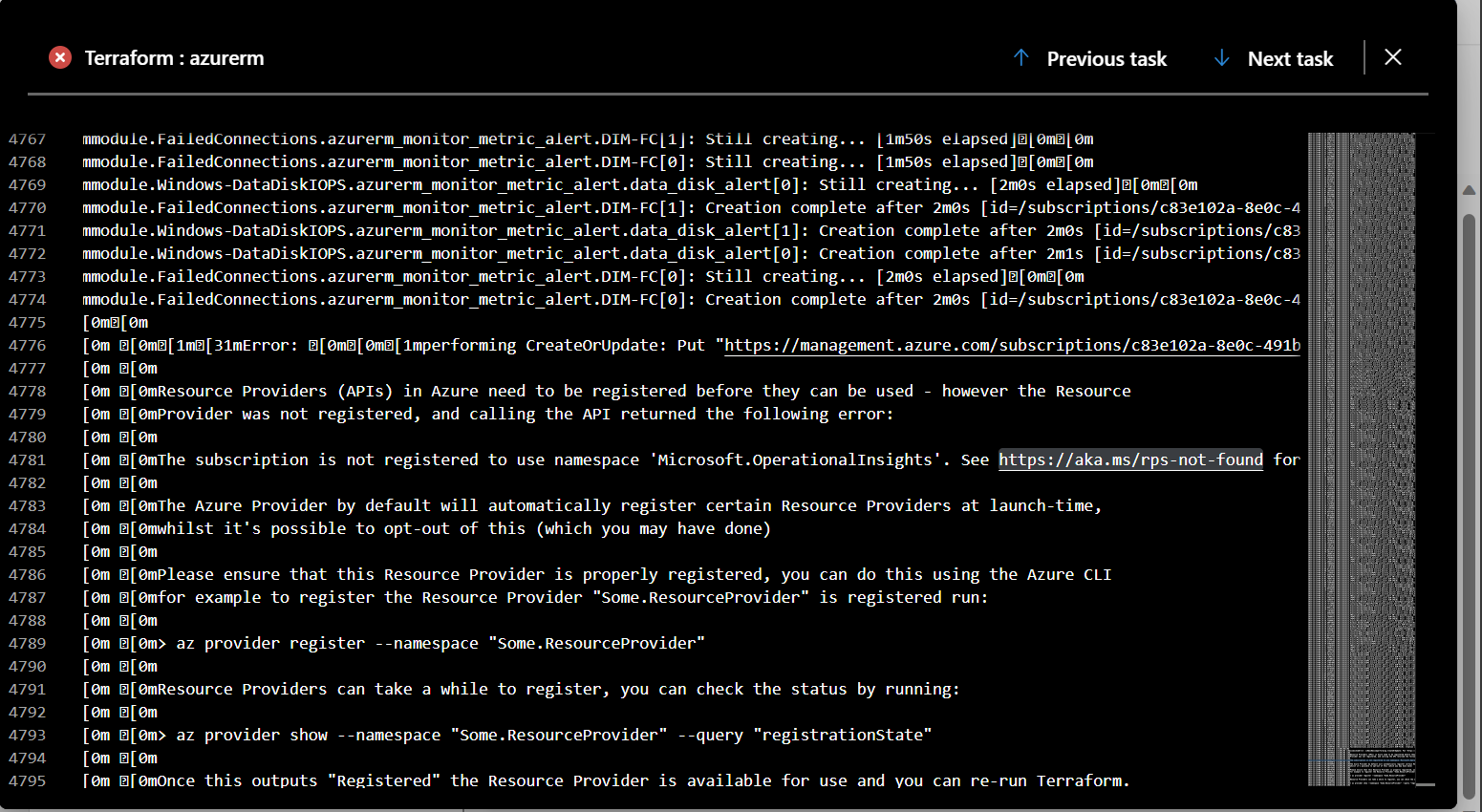
* [azurerm\_monitor\_activity\_log\_alert | Resources | hashicorp/azurerm | Terraform | Terraform Registry](https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/monitor_activity_log_alert)

It Manages an Activity Log Alert within Azure Monitor.

# Known Error Database (KEDB)

**Issue faced during the monitoring deployment phases**

* In case of azurerm\_monitor\_metric\_alert: When the evaluation frequency is 5, window\_size can be ignored. If evaluation frequency is greater than 5, then window size should be added which should be equal to evaluation frequency.
* Alerts are currently not supported with multi resource level for database servers, load balancers and front door & CDN profile.
* If the subscription is not registered to use the namespace, then you will get the below error.



* State file should be locked as to avoid/prevent the deletion. It should have do not delete and read only lock.

# FAQ’s

# Limitations-Terraform and Azure

* No error handling. This implies that we cannot utilize try-catch in the manner we do in other languages.
* A few things are prohibited from import.
* There is no way to roll back. As a result, we must delete everything and re-run if necessary.
* Terraform null\_resource does not have a state which means it will be executed as soon as you run $ terraform apply command but no state will be saved.