

## Terraform contd

- Lets add the storage account declaration

```
terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "3.102.0"
    }
  }
}

provider "azurerm" {
  features {

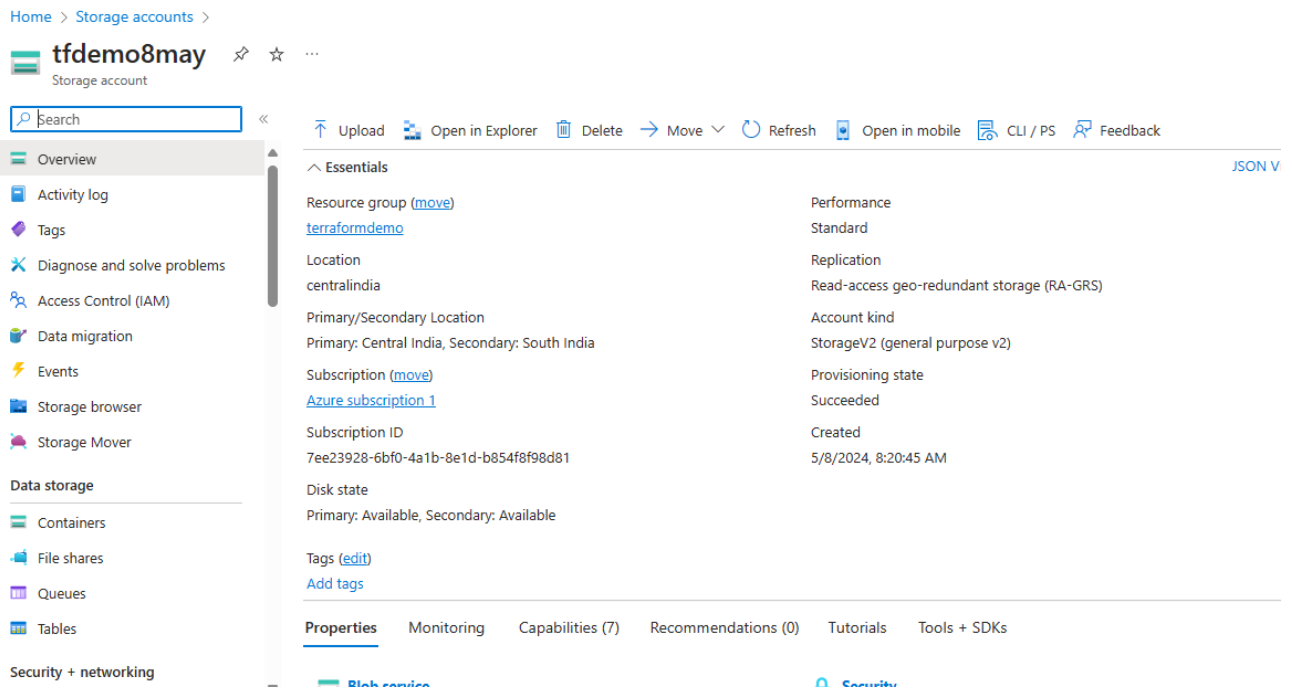
  }
}

resource "azurerm_resource_group" "storage" {
  name      = "storage"
  location  = "Central India"
}

resource "azurerm_storage_account" "example" {
  name                            = "fromtfmay24"
  resource_group_name            = azurerm_resource_group.storage.name
  location                       = azurerm_resource_group.storage.location
  account_tier                   = "Standard"
  account_replication_type       = "GRS"

  tags = {
    environment = "staging"
  }
}
```

- Now validate, format and apply to create infra



- [Refer Here](#) for the changeset
- Now destroy the resources

## Hashicorp Configuration Language

- Block syntax

```
block "name" {
  arg1 = value1
  ..
  argn = valuen
}
```

- Provider Block: This defines where to create resources. [Refer Here](#)

```
provider "<name>" {
  arg1 = value1
  ..
  argn = valuen
}
```

- Terraform block: [Refer Here](#) for official docs, we use terraform block to specify which provider version can be used and also which terraform version can be used
- Version constraints [Refer Here](#)
- Datatypes: [Refer Here](#)
- Resource block: [Refer Here](#) for official docs

```
resource "<type>" "<name>" {

}
```

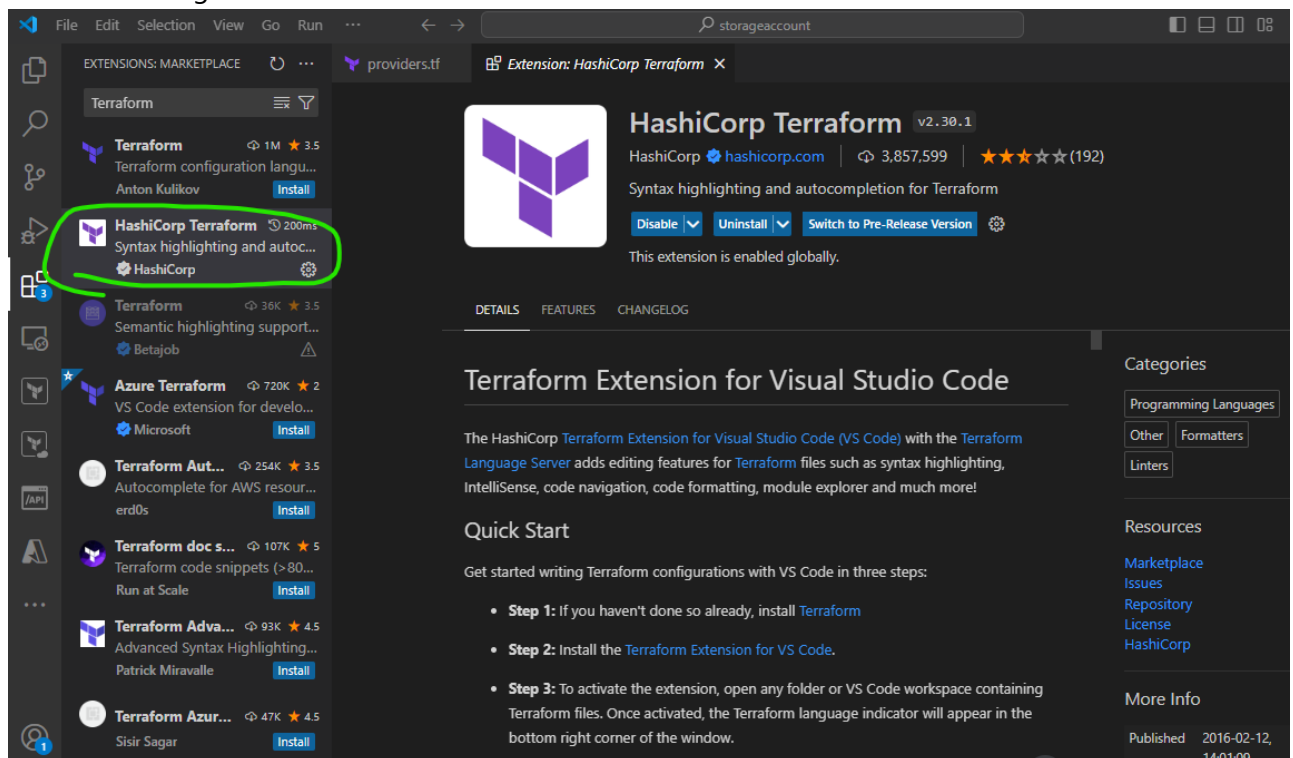
- type defines type of the resource generally type will be in the form of `<provider>_resource => azurerm_resource_group`
- name here is not the iname of the resource that will be generated in provider rather this is reference or alias to resource for usage with in template.

```
resource "azurerm_resource_group" "noname" {
  name = "luckyname" # this is name of resource in azure
}
```

- Generally when you create names use underscores rather than hypens

## Terraform Visual Studio Code Setup

- We will be using Terraform Extension in visual studio code



## Terraform template basic structure

- Create a folder with some meaningful name.
- In this we will be creating minimum 4 files all the time
  - main.tf: The major infra to be created
  - providers.tf: This file will have provider and terraform configuration
  - inputs.tf/variables.tf: This file will have variables
  - outputs.tf: this file has outputs to be shown to the user

## Configuring Azure Provider

- Every Provider in terraform needs credentials. Terraform provides various options to authenticate to Azure [Refer Here](#)
- We will be using the cli credentials
- To declare azure provider we need to create a provider block with name as azurerm

```
provider "azurerm" {  
  
}
```

- Every resource will have documentation of arguments and attributes

The screenshot shows the Terraform Registry page for the `azurerm_resource_group` resource. The page is titled "azurerm\_resource\_group" and includes a search bar on the left with the text "resource\_group". The search results show 12 matching results, with the first result being "azurerm\_resource\_group". The main content area displays the resource name and a description: "Manages a Resource Group." Below this, there are two "Note" boxes. The first note states: "Azure automatically deletes any Resources nested within the Resource Group when a Resource Group is deleted." The second note states: "Version 2.72 and later of the Azure Provider include a Feature Toggle which can error if there are any Resources left within the Resource Group at deletion time. This Feature Toggle is disabled in 2.x but enabled by default from 3.0 onwards." On the right side, there is a "ON THIS PAGE" section with links to "Example Usage", "Arguments Reference", "Attributes Reference", "Timeouts", and "Import". The "Arguments Reference" and "Attributes Reference" links are circled in green. At the bottom of the page, there is a cookie notice and buttons for "Manage Preferences" and "Dismiss".

- [Refer Here](#) for the changes done according to things learnt in class today