|  |
| --- |
| Infrastrcuture as Code |
| 5. øving i faget IaC |
| Forfattere:  Tor Ivar Melling  Eiere:  Forfatterne og Forskningsstiftelsen TISIP  Versjon: 1.0  Dato: 29. september 2020 |

# praktisk øving

I denne øvingen skal dere benytte Visual Studio Code til å jobbe videre med ARM Templates. Det ligger en videoer som dere skal benytte som ressurser for denne leksjonen. Her er det noe repetisjon fra sist, men det skader aldri.

*I denne øvingen skal dere vise hvordan en kan deploye ressurser til Auzre ved bruk av parameterfiler i Visual Studio Code. Test først med en enkel fil og utforsk deretter mulighetene med egne parameterfiler for forskjellige miljø, eksempelvis produksjon og utvikling. Forsøk også å legg ved en eller flere verdier i scriptet som tar presedens over verdien i parameterfilene.*

*Øvingsfiler ligger i egen mappe i its learning. Merk at det er både PowerShell og Bash-filer (Azure CLI).*

Det er ingen skriftlig leksjon denne gangen, men jeg anbefaler alle å søke opp ord eller utrykk som en er usikker på hva betyr. F.eks. Google: «what is azure cloud shell» om en lurer på hva dette er for noe.

For å kunne gjennomføre denne øvingen trenger dere en service principal som har rettigheter til å opprette ressurser i Azure tenanten som vi benytter og en bruker som har read only pålogging til Azure Portal – <https://portal.azure.com>

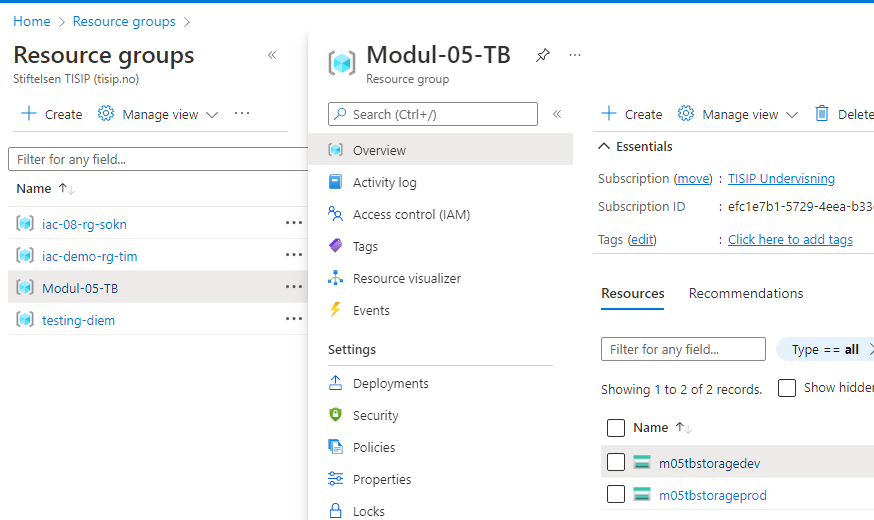
Service principal inneholder en ID og en Secret som er to lange strenger

Bruker med read only er en e-postadresse og passord.

Som levering på denne øvingen kan en levere en print screen av Azure Portal samt innholdet fra arm templaten og powershell-filen i et word dokument.

**HUSK**: Slett alltid ressursene etter at dere ferdig.

## Bilde fra Azure-portal



## Arm-Template

{

    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",

    "contentVersion": "1.0.0.0",

        "parameters": {

            "accountName": {

                "type": "string",

                "minLength": 3,

                "maxLength": 24,

                "metadata": {

                    "description": "Account name for storage account"

                }

            },

            "apiVersion": {

                "type": "string",

                "allowedValues": ["2015-05-01-preview", "2015-06-15", "2016-01-01", "2016-05-01", "2016-12-01", "2017-06-01", "2017-10-01", "2018-02-01", "2018-03-01-preview", "2018-07-01", "2018-11-01", "2019-04-01", "2019-06-01", "2020-08-01-preview", "2021-01-01", "2021-02-01", "2021-04-01", "2021-06-01", "2021-08-01"],

                "defaultValue": "2015-06-15",

                "metadata": {

                    "description": "Selection for Azure API-version"

                }

            },

            //Defines possible storage configurations

            "storageSKU": {

                "type": "string",

                "defaultValue": "Standard\_LRS",

                "allowedValues": ["Standard\_LRS","Standard\_GRS","Standard\_ZRS","Premium\_LRS"],

                "metadata": {

                    "description": "Gives select options for storage configuration on storage account"

                }

            }

        },

        //"variables": {

          //  "accountName": "[concat(parameters('accountName'), uniqueString(resourceGroup().id, subscription().tenantId))]"

        //},

        "resources": [

            {

                "name": "[parameters('accountName')]",

                "type": "Microsoft.Storage/storageAccounts",

                "apiVersion": "[parameters('apiVersion')]",

                "location": "[resourceGroup().location]",

                "tags": {

                    "displayName": "storageaccount1"

                },

                "properties": {

                    "accountType": "[parameters('storageSKU')]"

                }

            }

        ],

        "outputs": {

            "Name": {

                "type": "string",

                "value": "[parameters('accountName')]"

            }

        }

}

## Parameter-fil (Dev)

{

    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentParameters.json#",

    "contentVersion": "1.0.0.0",

    "parameters": {

        "accountName": {

            "value": "M05TBStorageDev"

        },

        "storageSKU": {

            "value": "Standard\_LRS"

        },

        "apiVersion": {

            "value": "2015-06-15"

        }

    }

}

## PowerShell

#Variables

$ResourceGroup = 'Modul-05-TB'

$ResourceDirectory = 'C:\IAC-TISIP-2022\Modul-05'

$UserInput = $null

#Create Azure resource group

New-AzResourceGroup `

    -Name $ResourceGroup `

    -Location 'West Europe' `

    -Force

#Create group deployment

New-AzResourceGroupDeployment `

    -Name 'M-05-TB-DEV' `

    -ResourceGroupName $ResourceGroup `

    -TemplateFile "$ResourceDirectory\Modul-05.json" `

    -TemplateParameterFile "$ResourceDirectory\Modul-05-dev.parameters.json"

New-AzResourceGroupDeployment `

    -Name 'M-05-TB-PROD' `

    -ResourceGroupName $ResourceGroup `

    -TemplateFile "$ResourceDirectory\Modul-05.json" `

    -TemplateParameterFile "$ResourceDirectory\Modul-05-prod.parameters.json"

#Gets user-input y/n for deletion of resource

Write-Host -ForegroundColor Cyan 'Resource Created. Verify resource before continuing...'

[String]$UserInput = Read-host -Prompt 'Enter anything and press enter when you want to remove resource'

if ($UserInput -eq 'y' -or 'yes') {

    Remove-AzResourceGroup -Name $ResourceGroup -Force -Verbose

}

# TEORETISK ØVING

* Hva tror du er fordelen ved å ta i bruk parameterfiler ved utrulling av ressurser i Azure?

Fordelene ved å bruke parameterfiler begynner man å se fordeler av når malene begynner å bli store og krever mange verdier.

Det hjelper også med oversiktligheten og gjør det enklere å bruke og forstå. Man kan også se fordelen ved at noen andre evt. skal ta i bruk mal og parameter-filene hvor det da blir enklere for dem å bruke og forstå uten for mye innsikt i det fra før.

Lykke til med øvingen og faget