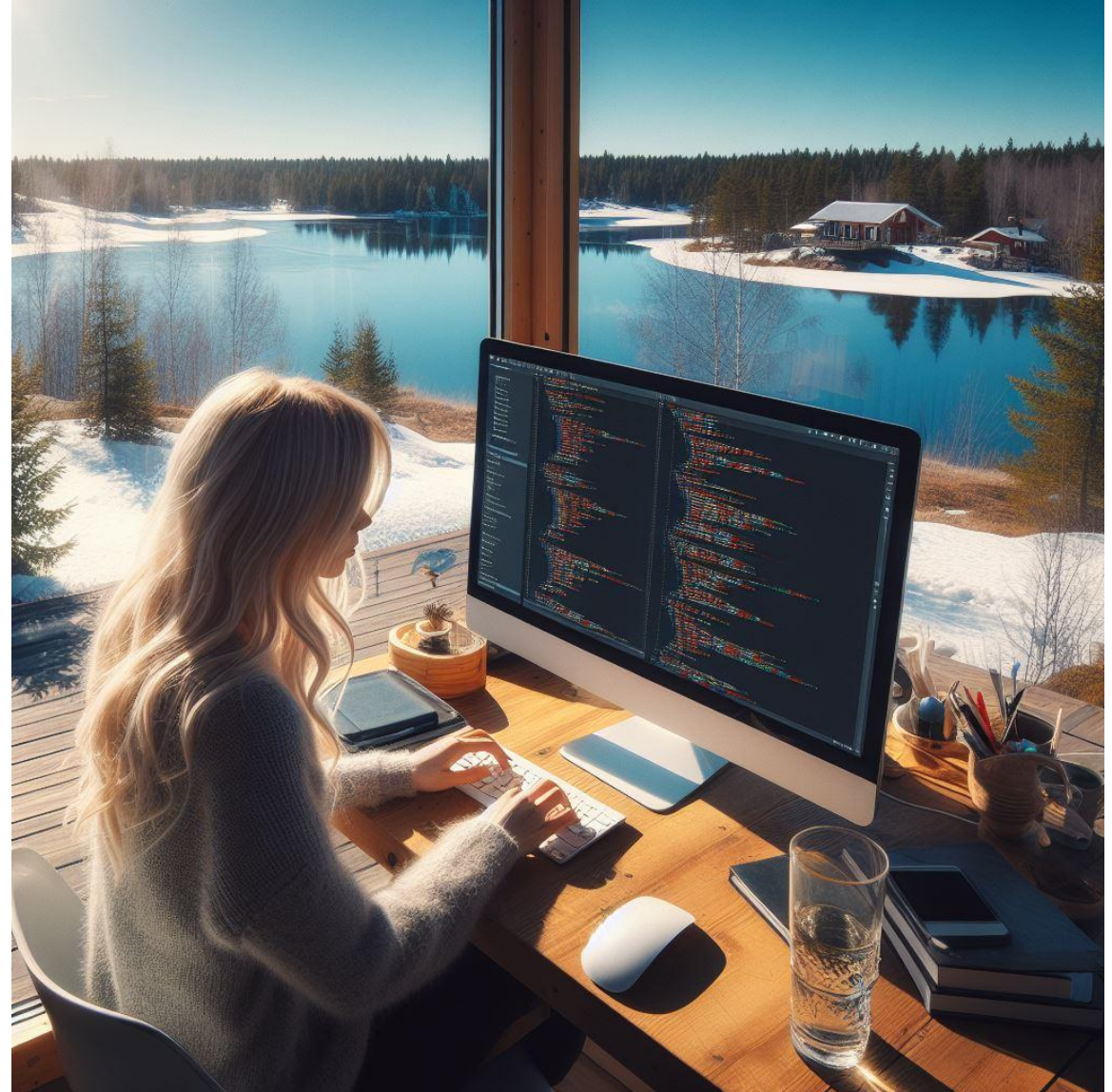




# Azure IaC Training



# Agenda

---

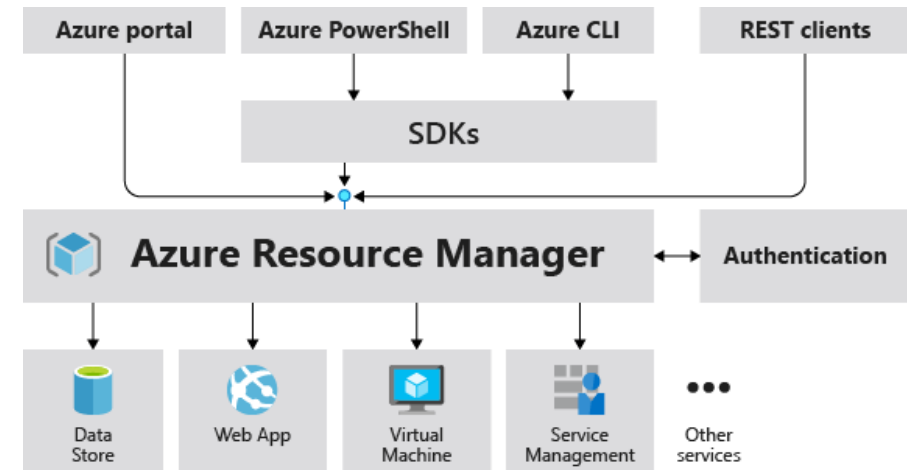
- Azure Resource Manager (ARM)
- Connecting to Azure
- ARM Templates
- Bicep
- Terraform

# Learning objectives

- Understand Azure Resource Manager
- Capability to read ARM-template
- Use and modify bicep-template
- Understand Terraform
- Optional: Create and use terraform-template

# Azure Resource Manager

- Engine behind everything
- Handles requests and converts everything to ARM-template
- Checks permissions



# Azure Deployment types

Subscription deployment



Resource group deployment



# ARM Template

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-  
01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "parameters": {},  
  "functions": [],  
  "variables": {},  
  "resources": [],  
  "outputs": {}  
}
```

# Bicep Template

```
param <name> <type> = <default value>
var <name> = <value>
resource <name> <type>@<API-version> = {
    name: <resource name>
    ...
}
output <name> <type> = <value>
```

# Login

```
Connect-AzAccount -Tenant mytenant.onmicrosoft.com
```

```
az login --tenant mytenant.onmicrosoft.com
```



# Subscription context

```
Get-AzContext
```

```
Set-AzContext -Subscription <GUID>
```

```
az account show
```

```
az account set --subscription <GUID>
```

# Subscription deployment

```
New-AzDeployment -Location swedencentral -TemplateFile  
.\subDeployment.bicep -TemplateParameterFile  
.\subDeployment.bicepparam
```

```
az deployment sub create --location swedencentral --  
template-file subDeployment.bicep --parameters  
subDeployment.bicepparam
```

```
az deployment sub create --location swedencentral --  
template-file subDeployment.bicep --parameters  
'@subDeployment.parameters.json'
```

# Resource group deployment

```
New-AzResourceGroupDeployment -ResourceGroupName rg-  
example1-bicep -TemplateFile .\storageDeployment.bicep -  
TemplateParameterFile .\storageDeployment.bicepparam
```

```
az deployment group create --resource-group rg-example1-  
bicep --template-file .\storageDeployment.bicep --  
parameters .\storageDeployment.bicepparam
```

```
az deployment group create --resource-group rg-example1-  
bicep --template-file .\storageDeployment.bicep --  
parameters '@storageDeployment.parameters.json'
```






# Deployment modes

Complete

Incremental

---

# Bicep thoughts

- What-if is useless... 
- Scripting capabilities are.... 
- Great resource support (straight from ARM)
- Supports newest technologies immediately
- Reduces a lot of code
- Easy to learn and use 



# Terraform fundamentals

- Adds another abstraction layer to Azure
  - Handles the whole environment like **Complete**-mode in ARM/bicep
  - Deployment scope are .tf-files that you have in the working directory
-



# Terraform fundamentals

- Multi-cloud support means that you have to understand each cloud separately and use only the same syntax on .tf-files
  - Stores deployment state to state-file that **MUST** be secured
  - Loose resource API version management
-

# ARM Template

```
variable <name> {  
    type          = <type>  
    default       = <default value>  
}  
  
resource <resource provider type> <name> {  
    name          = <resource name>  
    ...  
}  
  
output <name> {  
    value         = <value content>  
}
```



# Terraform run

```
terraform init
```

```
terraform plan -var-file="myVariables.tfvars"
```

```
terraform apply -var-file="myVariables.tfvars"
```

# Bicep thoughts

- Great validation support
- Scripting capabilities 👍
- Terraform plan 💖
- Breaks Azure's concept of resource versions
- State file makes an *extra layer* for status of environment and might include sensitive data in plain text
- New capabilities are not supported immediately
- Multi-cloud is not a real reason

# IaC template schemas

- [Azure resource reference - Bicep, ARM template & Terraform AzAPI reference | Microsoft Learn](#)