# DAY-3

## ARM TEMPLATE PARAMETERS:

Parameters

- Definition: Parameters are placeholders in an ARM template that allow you to input values at deployment time. They enable you to customize the deployment of resources without modifying the template itself.
- Example: In a storage account template, you might define a parameter for storageAccountName which can be set to different values depending on the environment (e.g., dev, test, prod).

Ex: file.myapp.json

```
"parameters": {
  "storageAccountName": {
    "type": "string",
    "metadata": {
      "description": "Name of the storage account"
    }
  }
}
```

Parameter Files:

- Definition: A parameter file is a separate JSON file that contains the values for parameters defined in an ARM template. It allows you to provide customizable inputs for your deployment without changing the main template.
- Structure: A parameter file only includes the parameter names and their respective values, omitting type definitions or metadata.

Ex: file.parameter.json

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentParameters.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
```

```
    "storageAccountName": {
      "value": "myuniquestorageaccountname"
    }
  }
}
```

Uses of Parameter Files in ARM Templates

1. Separation of Concerns: Parameter files decouple configuration from the template logic. This means you can modify the parameter values for different environments (development, testing, production) without altering the core template structure.
2. Ease of Management: When working in teams or planning multiple deployments, using parameter files helps maintain consistency and reduces the risk of errors retyping values directly in the template.
3. Flexibility: You can create multiple parameter files (e.g., parameters-dev.json, parameters-prod.json) to suit different deployment scenarios, making the ARM templates versatile for various environments.
4. Reusability: Templates and parameter files can be reused across different projects or teams, enhancing productivity and standardization in deployments.
5. Streamlined Deployments: Using parameter files allows for a cleaner command line interface when deploying templates, enabling you to pass all required parameters in a single file rather than specifying them individually as command-line arguments.

## Communication between ARM templates and ARMP (parameter files):

During the Deployment Process:

- When you deploy an ARM template using Azure PowerShell or Azure CLI, you can specify the ARMP file as a parameter. The deployment command retrieves the corresponding parameter values from the ARMP file and substitutes them into the ARM template during execution.
- For example, an Azure CLI command might look like this:
-

```
az deployment group create \
 --resource-group MyResourceGroup \
 --template-file MyTemplate.json \
 --parameters @MyParameters.json
```

My observations:

1.We should not define parameters first in the parameter {}. After defining in the resource only we need to define in the parameter. Other wise it will show the error.

2.my vs code error. The error message you're seeing ("Unknown word.cSpell") suggests that the code spell checker (likely part of your IDE or editor, such as Visual Studio Code) is flagging the term "rutwik" as a spelling error because it doesn't recognize it as a valid word. Its not the arm parameter file error.

3. The parameters file should only include the parameter names along with their values. No Types or Metadata. You do not include the parameter types, descriptions, or any metadata in the parameters file. This separation allows you to manage configuration separately from your template definition, enabling easier updates and different configurations for various environments.