

# Building a Web Catalog

For your private Bicep Registry

Simen Østensen

Global Azure Meetup Norway - Bergen - 18. April 2024



Global Azure 2024  
Meetup

April 18th 2024 | 12:00 |  
Kanalveien 11, Bergen

Azure  
User group  
Norway



# Introduction

**What to expect from this session?**

**Who am I?**



## The Problem

Documentation problem

Public vs Private Registry

## The Tools

PSDocs

MKDocs

## Live Demo

Publish module to ACR

Automate the documentation with PSDocs

Implement MKDocs

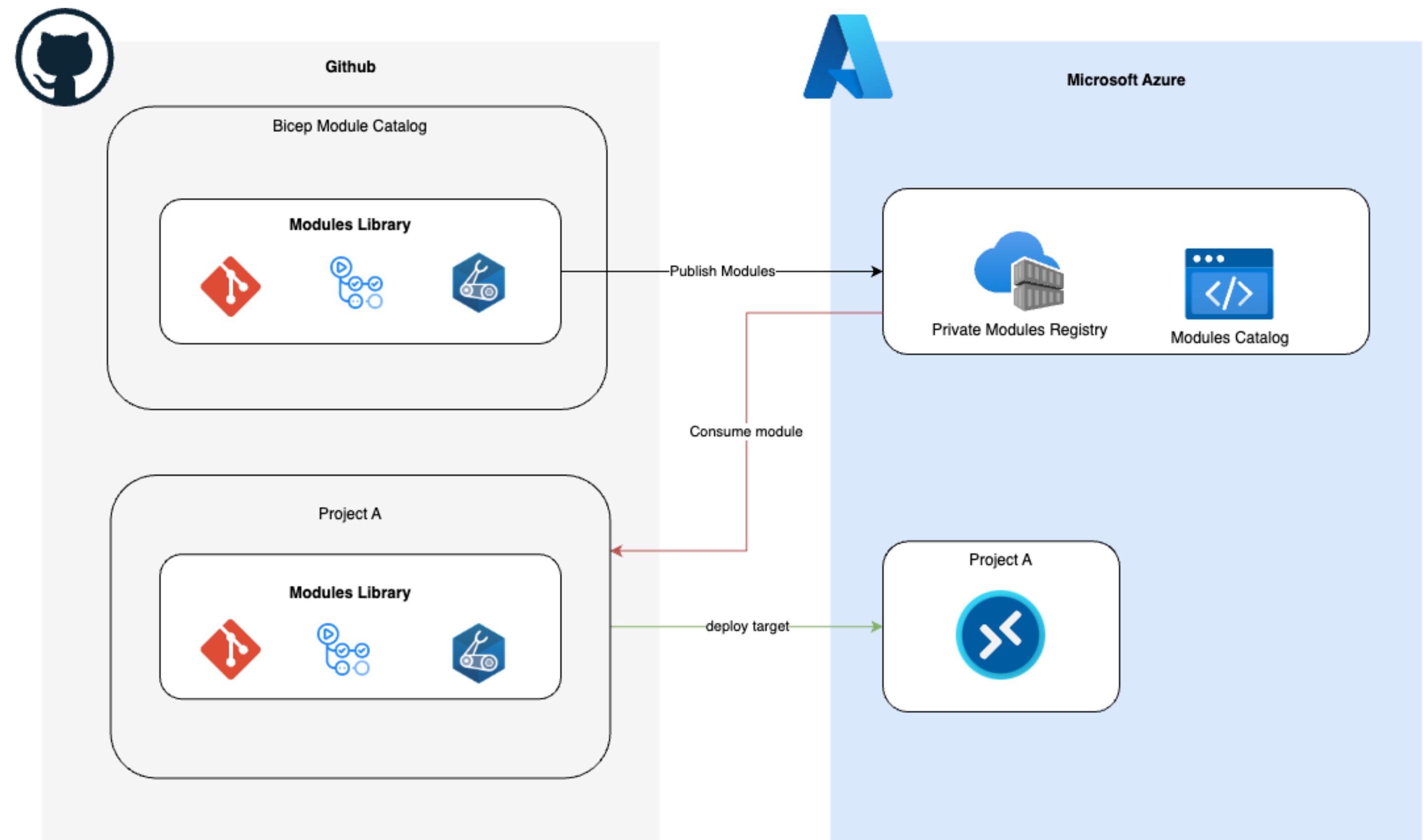


**Simen Østensen**

*Senior Cloud & DevOps Engineer  
Devoteam M Cloud*

# The Problem

- Project background
  - Automation library for Azure Virtual Desktop
  - Ended up with around 70 modules
- Initial challenges with module documentation
- Public or Private registry



# Azure Verified Modules

## Public registry for IaC Modules

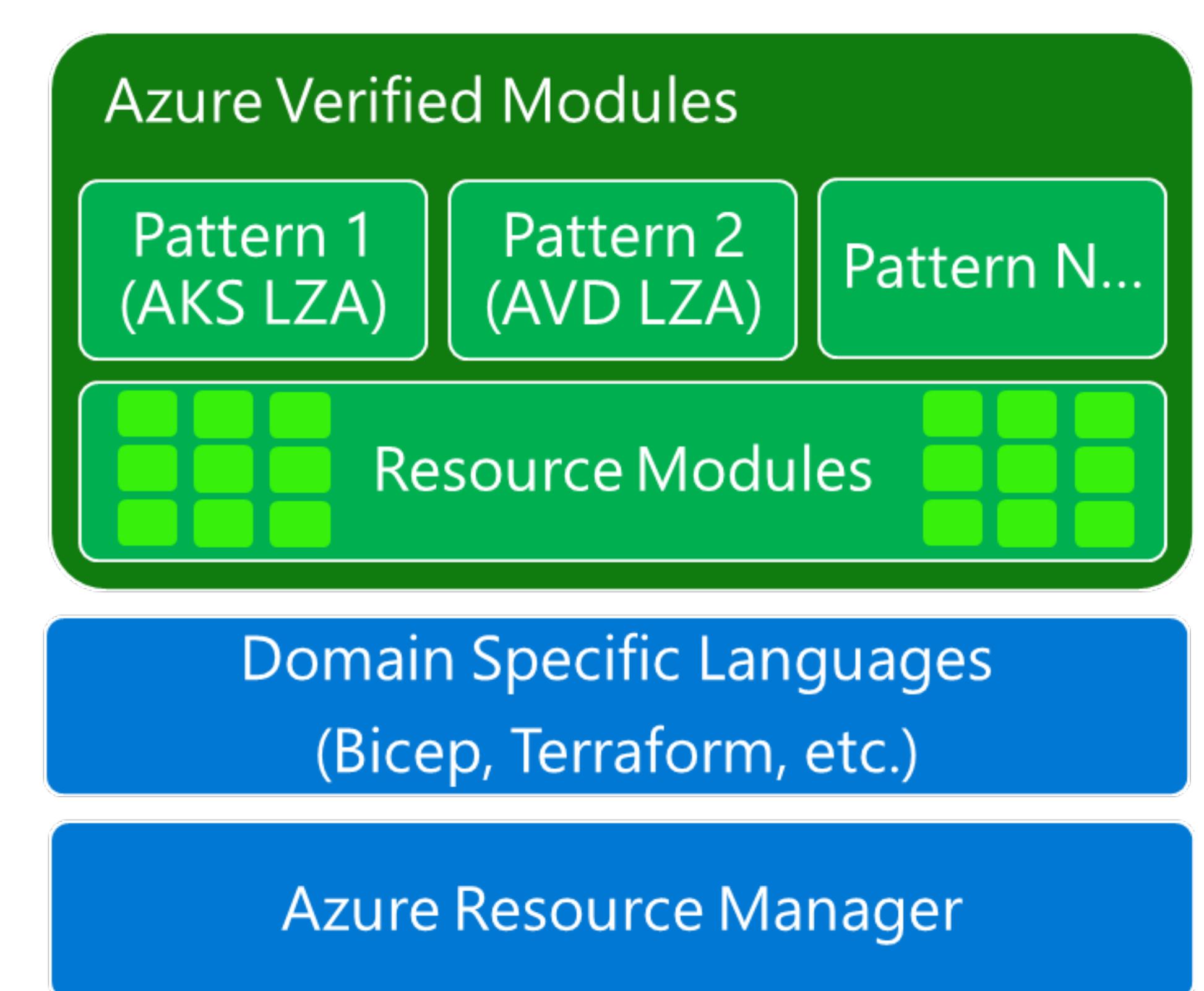
- Modules for both Bicep & Terraform
- Two classifications of modules (across both languages)
  - Resource Modules
  - Pattern Modules
- Benefits:
  - **Microsoft is taking the heavy lifting:** You don't have to maintain the modules
  - **Quality Assurance:** Each module is thoroughly tested
  - **Community Support:** Leveraging collective insights enhances module reliability and functionality

We got you covered!



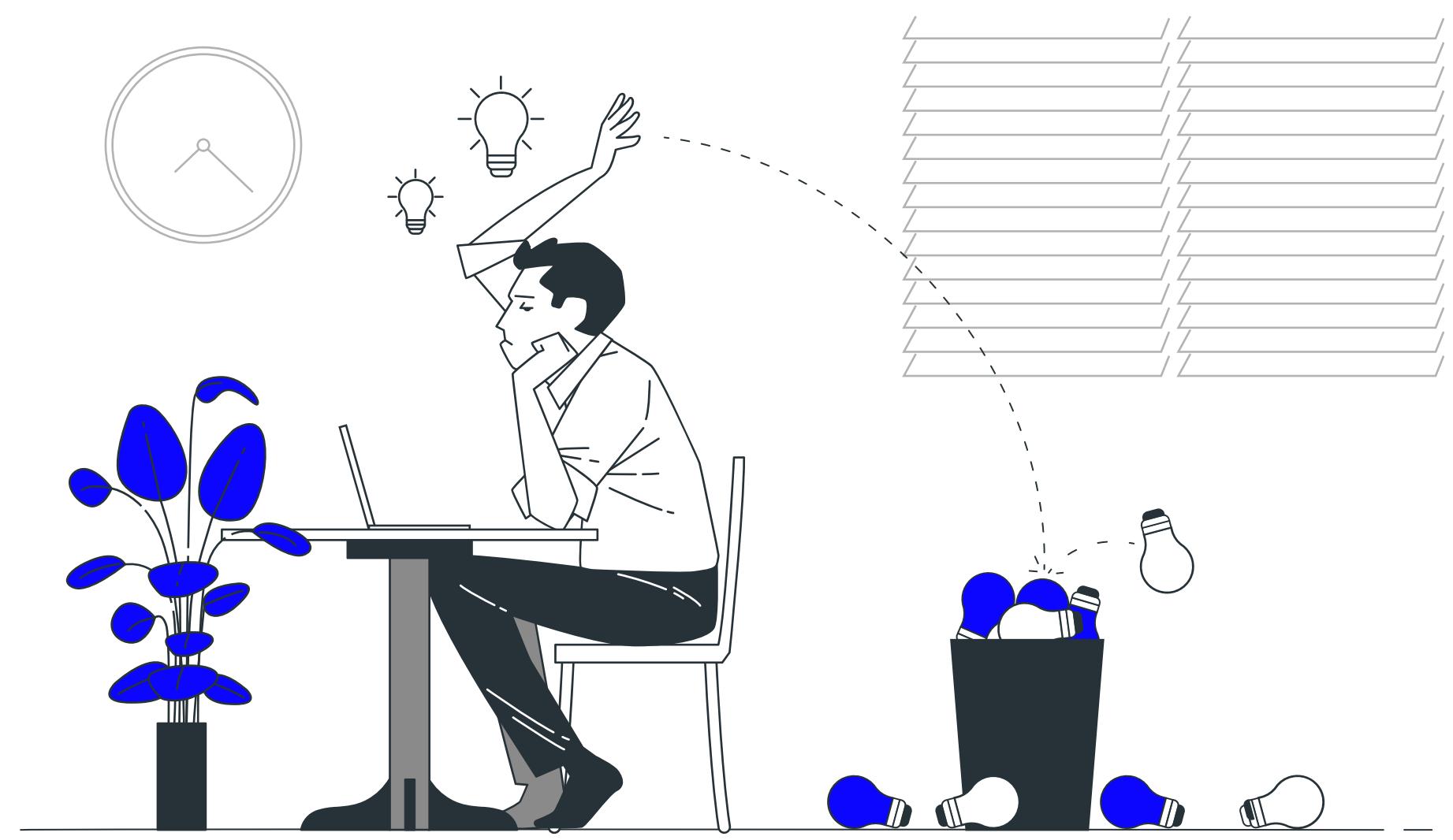
# Why should you consider still using a private registry then?

- A lot of overhead with hosting your own Azure Container Registry
- Build **pattern modules** -> Aligned with your company standards and regulations
- Reducing dependencies to fix issues from Microsoft



# Solution Exploration

- 70 modules to create documentation for
- Looked at various tools:
  - PSDocs for generating markdown files
  - Hugo
  - Jekyll
  - Mkdocs



# Solution

- PSDocs - To automate the process of creating markdown for our Bicep code
- MKDocs - To build our web catalog
- Azure Static Web App - For hosting our application



# Technical Deep Dive

- Tool Explanations
  - PSDocs
  - MKDocs
- Live Demos
  - Part 1: Deployment of Modules
  - Part 2: Automated Documentation with PSDocs
  - Part 3: Setting up the Web Catalog with MKDocs

# PSDocs

Create in-depth documentation for your modules

- Generates markdown from Azure infrastructure as code (IaC) artifacts.

- Limitation:

- Supports only Azure Resource Manager (ARM) template files.

```
$bicepFilePath = "/modules/res/keyVault/deploy.bicep"  
  
# Convert Bicep file to ARM template  
$armTemplateFilePath = [System.IO.Path]::ChangeExtension($bicepFilePath, 'json')  
bicep build $bicepFilePath --outfile $armTemplateFilePath  
  
# Generate documentation for the  
Invoke-PSDocument -Module PSDocs.Azure -InputObject $armTemplateFilePath
```

# Output Simplest form

```
targetScope = 'subscription'

param name string
param location string = deployment().location

resource rg 'Microsoft.Resources/resourceGroups@2023-07-01' = {
    name: name
    location: location
}
```

## Outputs

### Azure template

### Parameters

Parameter name	Required	Description
name	Yes	
location	No	

### name

parameter required

### location

parameter optional

- Default value: [deployment().location]

### Snippets

#### Parameter file

```
JSON ▾
{
    "$schema": "<https://schema.management.azure.com/schemas/2015-01-01/deploymentParameters.json#>",
    "contentVersion": "1.0.0.0",
    "metadata": {
        "template": "test.json"
    },
    "parameters": {
        "name": {
            "value": ""
        },
        "location": {
            "value": "[deployment().location]"
        }
    }
}
```

# Customisation

- Uses metadata and the template structure to dynamically generate documentation
- Allows you to:
  - Include information with minimum effort
  - Use DevOps culture to author IaC documentation
  - Keep documentation up to date with your modules

Field	Scope	Type	Description
<code>name</code>	Template	<code>string</code>	Used for markdown page title
<code>summary</code>	Template	<code>string</code>	Used as a short description for the markdown page.
<code>description</code>	Template	<code>string</code>	Used for detailed description for the markdown page
<code>description</code>	Parameter	<code>string</code>	Used as the description for the parameter.
<code>example</code>	Parameter	<code>string</code> , <code>boolean</code> , <code>object</code> , or <code>array</code>	An example use of the parameter. The example is included in the JSON snippet. If an example is not included the default value is used instead.
<code>ignore</code>	Parameter	<code>boolean</code>	When <code>true</code> the parameter is not included in the JSON snippet.
<code>description</code>	Output	<code>string</code>	Used as the description for the output

# Output With metadata

```
metadata name = 'Resource Group'
metadata description = 'This module deploys a Resource Group.'

targetScope = 'subscription'

@description('Required. Name of your Resource Group')
param name string

@description('Optional. Location for your resource')
param location string = deployment().location

resource rg 'Microsoft.Resources/resourceGroups@2023-07-01' = {
    name: name
    location: location
}

@description('The name of the resource group.')
output name string = rg.name

@description('The resource ID of the resource group.')
output resourceId string = rg.id

@description('The location the resource was deployed into.')
output location string = rg.location
```

## Resource Group

This module deploys a Resource Group.

### Parameters

Parameter name	Required	Description
name	Yes	Required. Name of your Resource Group
location	No	Optional. Location for your resource

#### name

parameter required

Required. Name of your Resource Group

#### location

parameter optional

Optional. Location for your resource

- Default value: `[deployment().location]`

### Outputs

Name	Type	Description
name	string	The name of the resource group.
resourceId	string	The resource ID of the resource group.
location	string	The location the resource was deployed into.

### Snippets

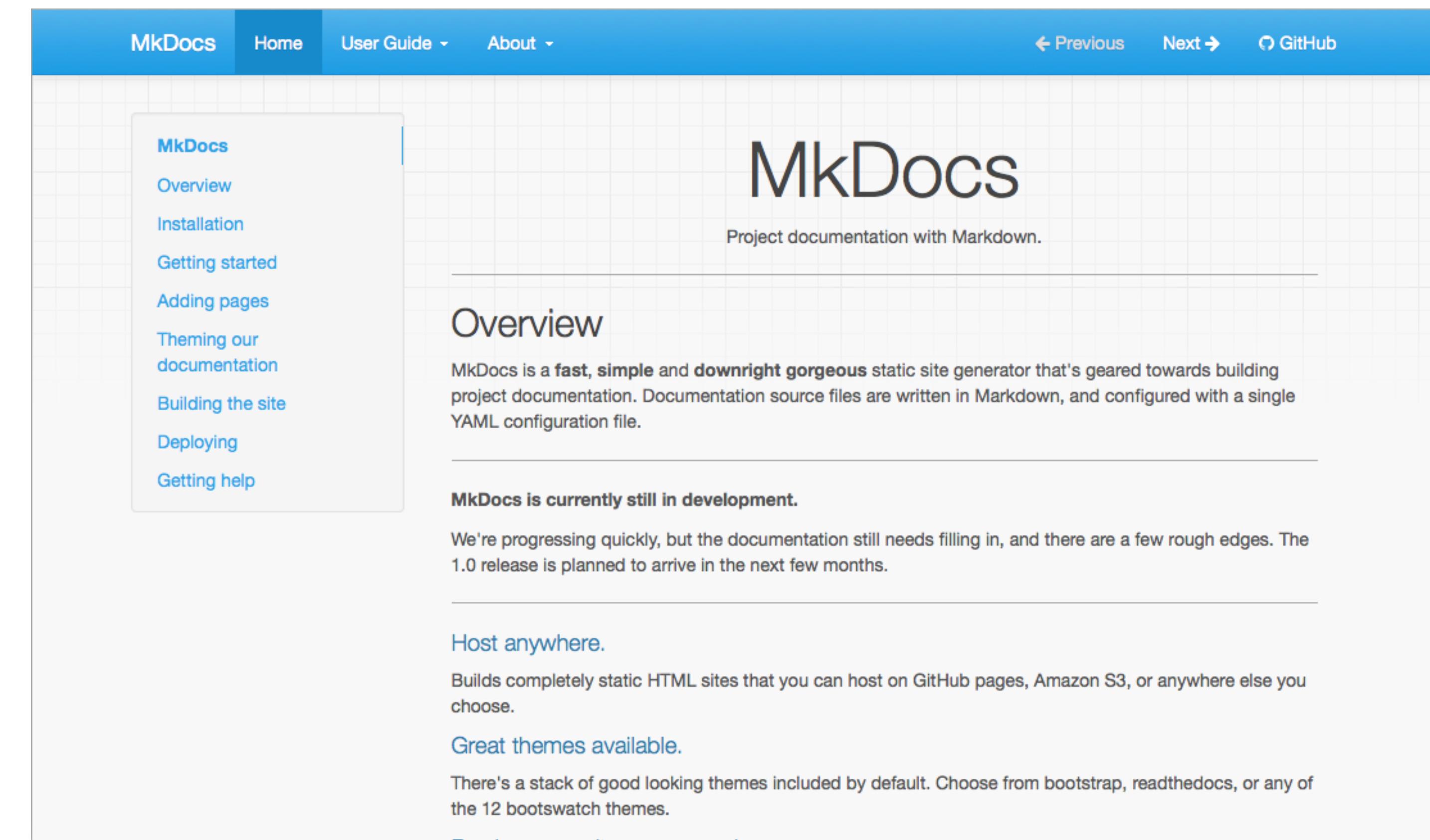
#### Parameter file

```
{
    "$schema": "<https://schema.management.azure.com/schemas/2015-01-01/deployme
ntParameters.json#>",
    "contentVersion": "1.0.0.0",
    "metadata": {
        "template": "test.json"
    },
    "parameters": {
        "name": {
            "value": ""
        },
        "location": {
            "value": "[deployment().location]"
        }
    }
}
```

# MkDocs

## Static site generator

- Build towards project documentation
- Written for Markdown
- Configured with a single YAML config file
  - Themes
  - Plugins
- Host anywhere



# Material Design



## Theme & features

- Built-in search bar
- Code annotations
- Many other features ++

The screenshot displays the 'Bicep Module Library' website. At the top, there's a dark header with the title 'Bicep Module Library' and a search bar labeled 'Search'. Below the header is a navigation sidebar on the left containing a tree structure of module categories: HOME, Pattern modules, Resource modules, Key Vault, Network, Application Security Groups, Network Interfaces, Network Security Groups (with sub-items Main, Security Rules, Private Endpoints, Public IP Addresses, Route Tables, Virtual Networks), and Virtual Networks. To the right of the sidebar, the main content area is titled 'Module catalog' and contains a section titled 'Search for everything' with a red arrow pointing to the search bar. Below this is the 'Introduction' section, which includes a detailed description of the private module registry and its benefits, followed by a bulleted list of features: 'Suggesting Changes', 'Linking to Documentation', and 'Demonstrating Implementation'. A 'Note' box at the bottom of this section provides a note about curated modules for Azure Virtual Desktop. Further down, the 'Getting Started' section explains the difference between Pattern and Resource modules. On the far right, a vertical sidebar lists additional resources: Table of contents, Introduction, Getting Started, Use a Bicep Module, Alias, Versioning, and Additional Resources. A red arrow points from the text 'Auto generated Table of contents' to this sidebar.

Module catalog

Search for everything

Introduction

To share Bicep modules within our organization, we have established a private module registry, hosted on Azure Container Registry. This ensures that our cloud resources are deployed consistently and reliably across multiple projects. Our private registry enables the secure sharing of internally developed modules, enhancing collaboration while maintaining strict security standards.

Importantly, each module within our registry undergoes rigorous testing with PS-Rule. Leveraging the principles of the Azure Well-Architected Framework (WAF), PS-Rule for Azure plays a critical role in our development process by:

- **Suggesting Changes:** Offering actionable recommendations to enhance the quality of our solutions.
- **Linking to Documentation:** Providing direct connections to relevant documentation, enabling our team to understand and apply best practices within our specific environment.
- **Demonstrating Implementation:** Showcasing examples in both Azure Bicep and ARM templates syntax, these demonstrations guide our developers in adopting the changes effectively.

**Note**

The modules currently published to our private container registry have been selectively curated for Azure Virtual Desktop deployments. However, the individual resource modules they are suitable for any other azure deployment.

Getting Started

The module catalog is divided into two main categories: Pattern modules (also known as composite modules) and Resource modules. Resource modules are standalone entities designed to manage specific Azure resources. In contrast, Pattern modules are constructed from multiple Resource modules, combined with custom modules to perform specific tasks.

Table of contents

Introduction

Getting Started

Use a Bicep Module

Alias

Versioning

Additional Resources

Organize your navigation bar

Auto generated Table of contents

# **Demo**

**Deployment of module, PSDocs & Mkdocs**

# Conclusion and Key Takeaways

- The importance of effective module management
- Choosing if the private registry is something for your company?
- Automating documentation with PSDocs
- Creating a web catalog with MKDocs

# Resources and Links

- Github - <https://github.com/SimenWO/bicep-module-library>
- PSDocs - <https://azure.github.io/PSDocs.Azure/overview/>
- MKDocs - <https://www.mkdocs.org/>
  - Material design - <https://squidfunk.github.io/mkdocs-material/>
- Azure Verified Modules - <https://azure.github.io/Azure-Verified-Modules/>

# Q&A

# Thank you!

**Simen Østensen**  
*Senior Cloud & DevOps Engineer  
Devoteam M Cloud*