Microsoft

SaaS Lab

Welcome to the

# SaaS Lab Program

saaslab@microsoft.com

Session 4

# Infrastructure as Code

From zero to Hero

This event will be recorded. Your name or other information may end up in the recording. If you do not wish to be recorded, please drop out of this session.

The event will start shortly

# Hello, meet your session contributors

## Nhi Tran

Cloud Solution Architect

About: Nhi is a Cloud Solution Architect with 18 years' experience in software development, system design, architecture and team management. Nhi has taken different role in her career. In Microsoft here, she is current focusing on cloud architecture with Azure and SaaS development.

nhitran@microsoft.com
https://www.linkedin.com/in/trankietnhi/

## Sajeetharan Sinnathurai

Cloud Solution Architect

About: About: With over 11 years of experience in the IT industry, Sajeetharan is a Cloud Solution Architect, an enthusiast in Cloud and Opensource. He mainly focus on channeling his knowledge into opensource projects and sharing it with the community by mentoring, writing blogs to help make the world a better and more developed place.

sasinnat@microsoft.com
https://www.linkedin.com/in/sajeetharan/

## Vorapat Nicklamai

Cloud Solution Architect

About: Vorapat (Guide) has been actively engaging enterprise customers and ISVs to help them with Azure architecture for the past years. He brought his software development and DevOps skills during his time as a site engineer of a high-transacting flight booking platform. Now he is expanding his DevOps journey into MLOps.

vnicklamai@microsoft.com
https://www.linkedin.com/in/vorapatnicklamai/

# Your feedback is important

Please help us improve this program by completing this short feedback form.

https://aka.ms/saaslabfeedback4

Microsoft

# If you'd like more help on your Azure modernization journey, please e-mail the SaaS Lab team

saaslab@microsoft.com

## Thank you for being part of the SaaS Lab Program

# Infrastructure as Code

**From Zero to Hero**

## Today's Discussion

- Challenge and Importance of Infrastructure as Code
- ARM template and resource deployment
- Demo
- Break poll
- IaC change process automation
- IaC best practice test, security validation and impact assessment in action
- Kahoot
- Q&A Reference

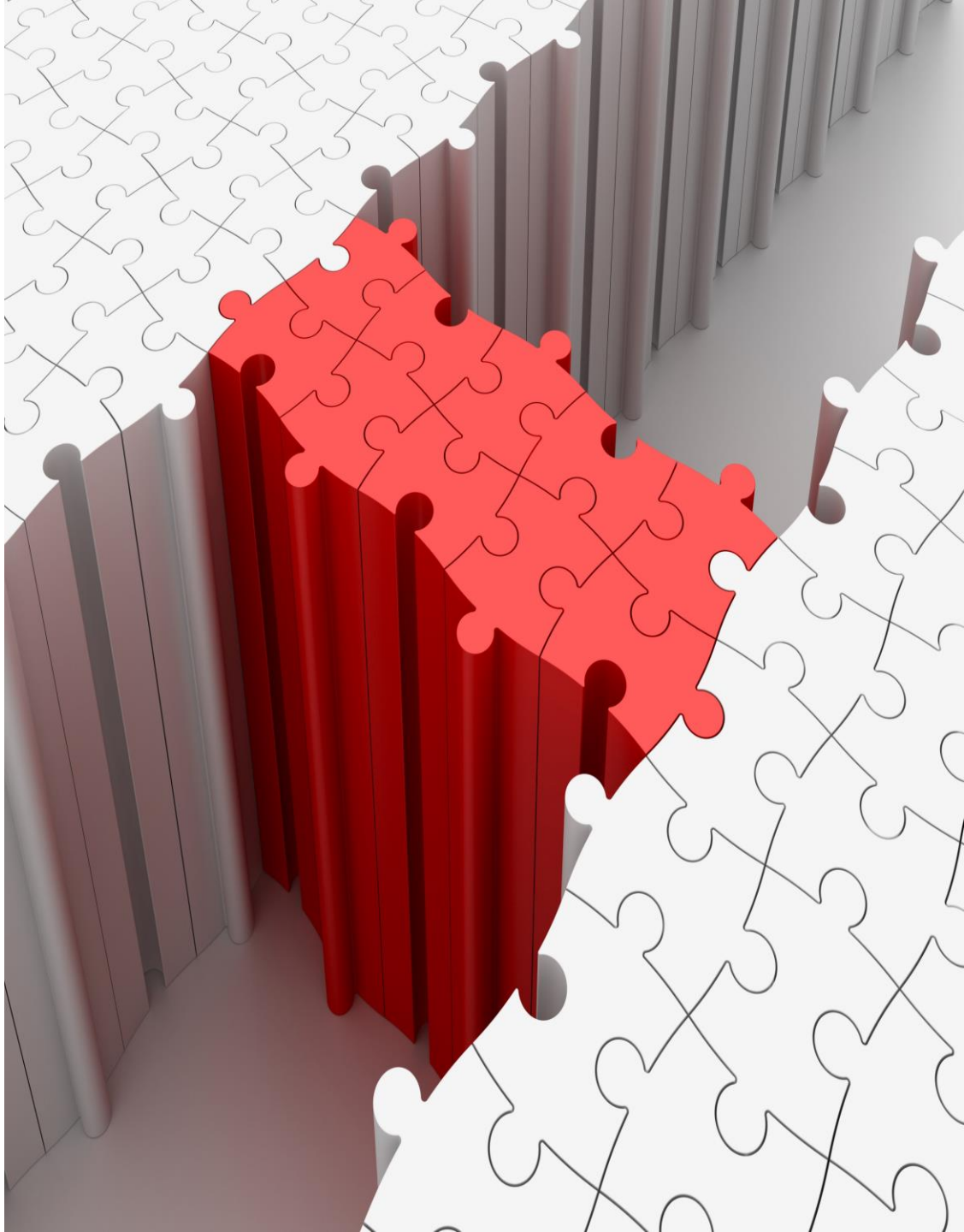
# Infrastructure Challenges for ISV

**Challenges:**

- Fast response to demand change
- Onboarding new tenant
- Keep your system healthy
- Meet required capacity
- New environment for innovation
- Compliance.

**Traditional Problems**

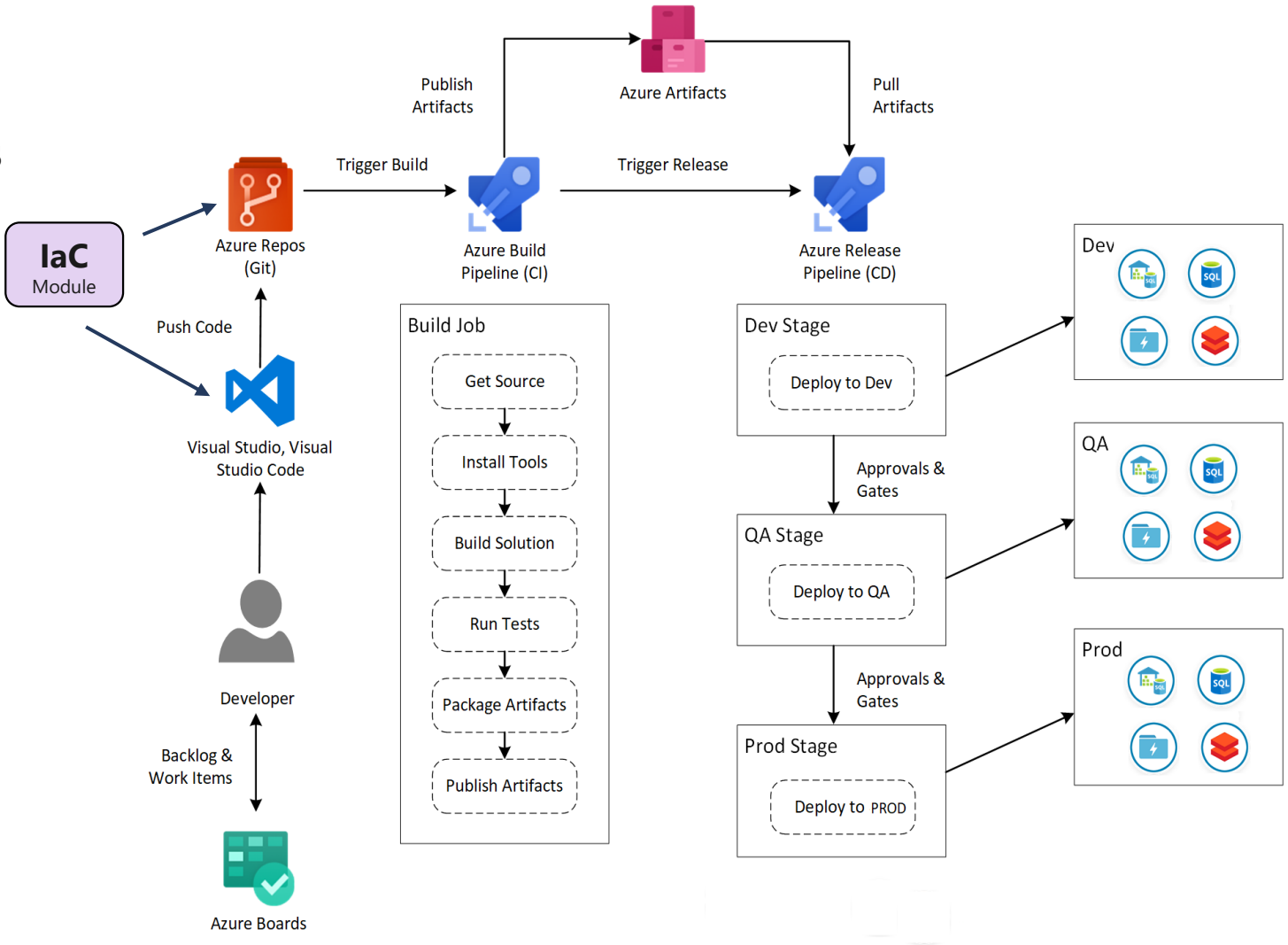
- Unexpected settings changed
- Low infrastructure visibility
- Slow/error-prone deployment
- Lack of collaboration



# Infrastructure as code

**Infrastructure as Code** treats the infrastructure **as a software system**, **applying software engineering practices** to implement and manage changes to the system in a **repeatable**, **structured** and **safe way**.

- Define deserved infrastructure
- Provision
- Configuration



# Benefit of Infrastructure-as-Code? The “ity’s” (mostly)

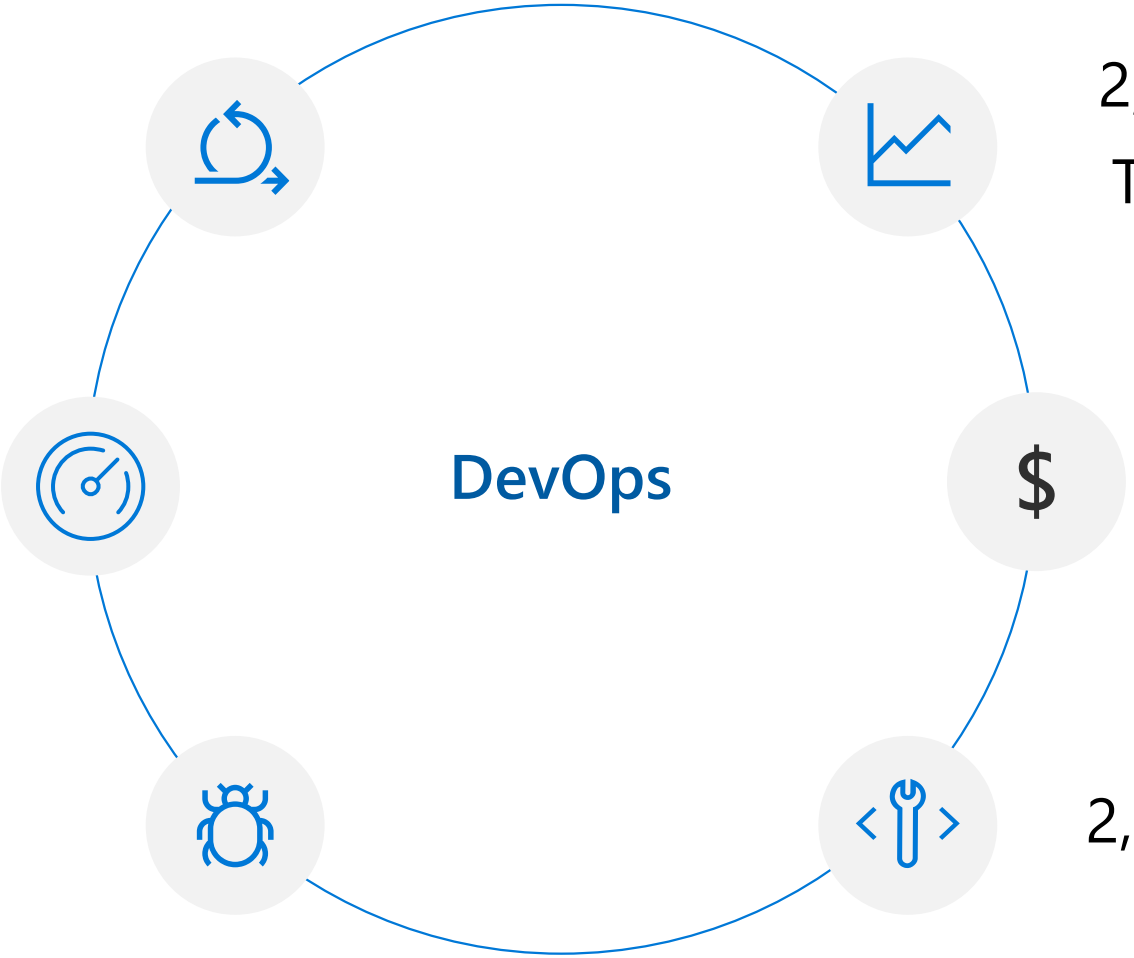
| Quality | Description |
|---|---|
| Confidence | Consistency behavior through automation, limited “Click-Ops”.<br>Systematic approach, limited human factors. |
| Repeatability | Identical test/dev cycles |
| Troubleshooting ability | Replica table elsewhere for troubleshooting |
| Recoverability | Easier to redeploy that to backup and restore.<br>Simple options for recovering systems elsewhere. |
| Auditability | Detailed tracking of changes with ability to rollback |
| Visibility | Limited ‘in box configuration’ if you have access to see the code, all configuration is visible. |
| Portability | Ability to deploy to other test/dev/sandboxed environments. |

# High Performance DevOps Companies Achieve…

46x Deployment Frequency

2,555x Faster Lead Time For Changes

**Faster Time to Market**




**Increased Revenue**

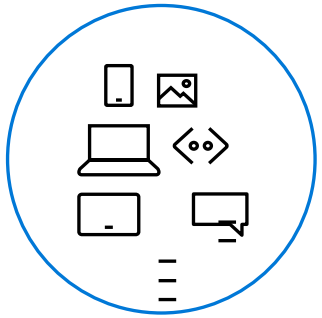
7x Lower Change Failure Rate

2,604x Faster Mean Time to Recover

Source: 2018 Accelerate: State of DevOps: Strategies for a New Economy." N. Forsgren, J. Humble, G. Kim. DevOps Research and Assessment (DORA)

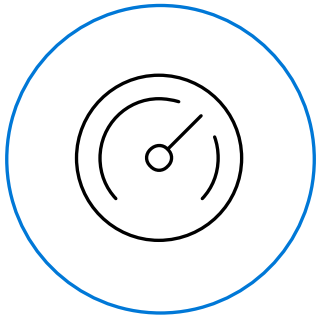
# Common Technologies of Infrastructure as Code

Some common **declarative** Infrastructure as Code technologies that can be used for different target systems.



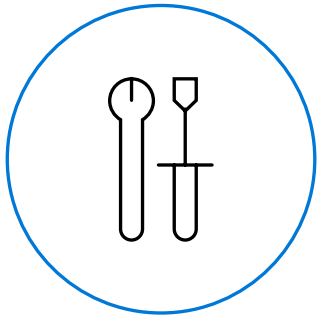
## Platform

- Azure Resource Manager (ARM) Templates
- Terraform
- VMWare Cloud Templates
- Pulumi
- Bicep (*)



## Config Management

- PowerShell DSC
- Ansible
- Chef
- Puppet
- Saltstack



## Other

- Kubernetes Manifests

# ARM template and resource deployment

# Benefits of using ARM Templates vs. other Infra as Code solutions

1 Resource **Provider coverage** from day one

*When a new Azure Resource is released there is an ARM Template available*

2 Deployments are a **tracked object** in the Azure Portal

*View the ARM Template deployment in the AzPortal, not the case for other infra as code solutions*

3 Pre-Flight **Checks**

*ARM Deployments have pre-flight checks to make sure the template will deploy successfully*

4 Azure Policy **Remediations**

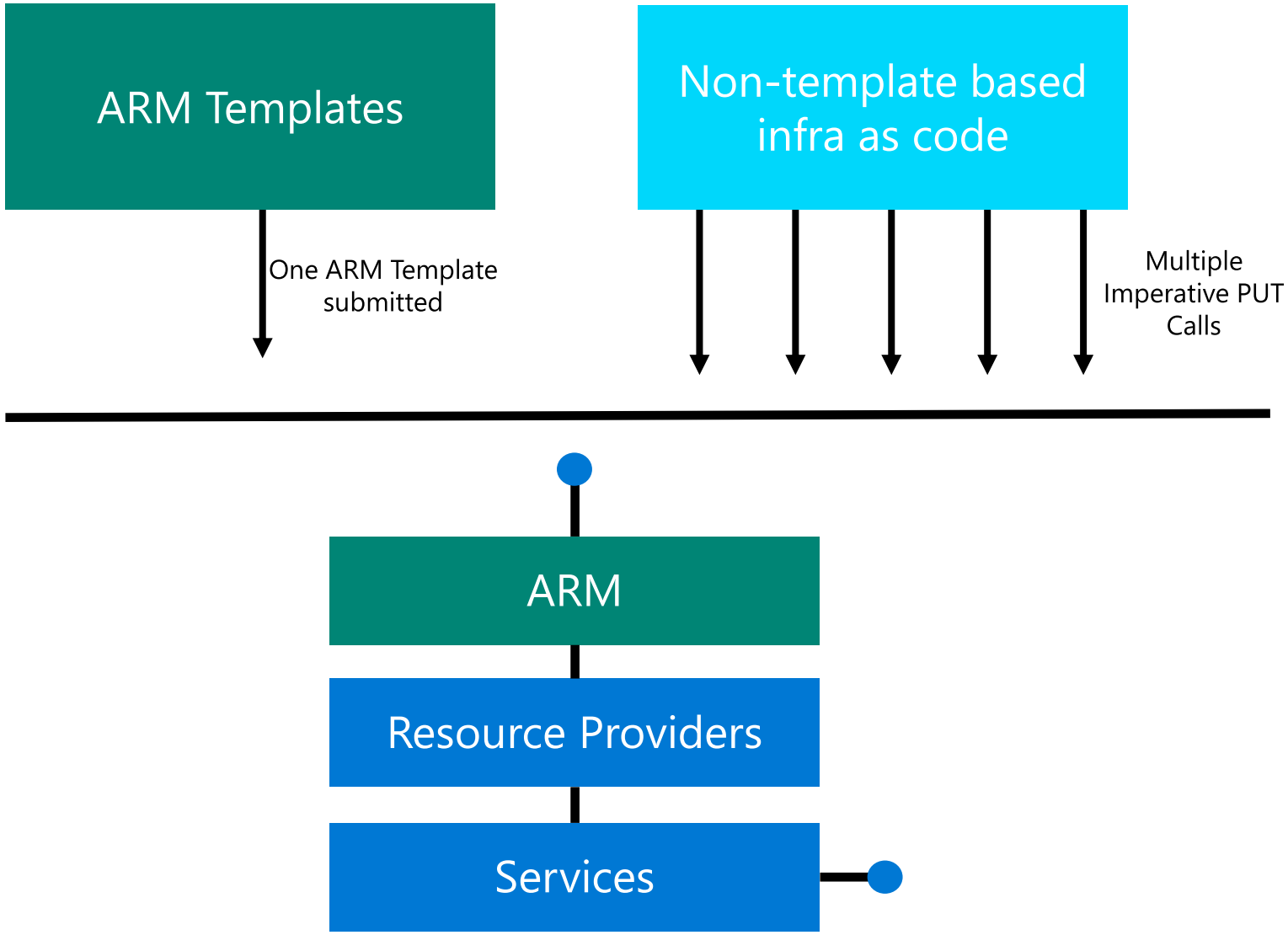
*If a customer is using Azure Policy the remediations performed for non-compliant resources are done through ARM Templates.*

5 **Concurrent** Executions

*3rd party IaC solutions do not allow current runs, plans or applies. No state management in ARM*

6 **Blueprint\Service Catalog**

*Ability to publish approved set of templates and reference architectures across your organization. ISO, NIST, HIPAA etc.*



# IT as Code narrowing the gap between developers and IT

## Cloud Center of Excellence (CCOE)

- Provision Tenant Infra
- Assign Policies
- Assign RBAC
- Create Subscriptions

## App Team/DevOps

- Create Pipelines
- Deploy App Infra
- Deploy Apps

**70%**

# of all resource created declaratively monthly via ARM Templates

Provision and manage the lifecycle of resources in a declarative way

## Declarative Approaches

1. Infra as Code
2. Policy as Code
3. Config as Code
4. Role based access as Code

## Capabilities Provided

### Day One and @ Scale

Resource Provider coverage from day one and ability to do large scale multi-region deployments

### Simplify Authoring

- Automated Template generation via Azure Portal
- VSCode extension with intellisense, snippets etc.

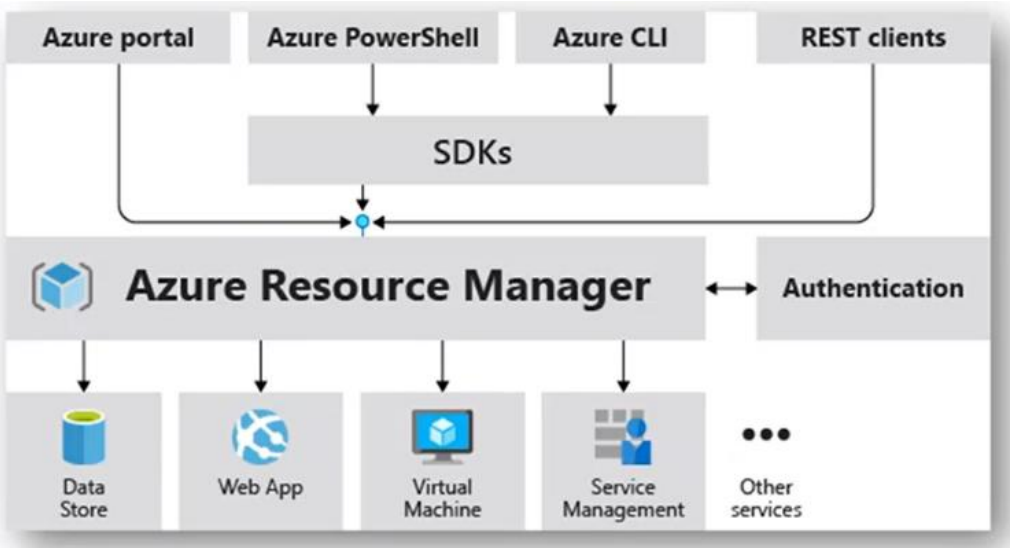
### Environment Setup

- Tenant, MG and Subscription level deployments
- One-click deployment solutions(Blueprints) to help meet regulations such as ISO, CIS, PCI, FedRAMP

### Integrated

- Azure Policy the remediations
- AzDevOps and GitHub tasks
- Provisioning flexibility with Terraform, Ansible, SNOW

# Understand ARM




- The Azure Resource Manager (ARM) is the deployment and management service for Azure.
- All Requests are handled by the same API providing consistent results regardless of the tools used to deploy
- ARM terminology
  - Resource (manageable item in Azure)
  - Resource Group (container holding resource)
  - Resource Manager Template (JSON file defining one or more resource to deploy)
  - Declarative syntax (define intention without sequencing commands to create)

# What are ARM Templates?

Declarative files for creating Azure resources in a reliable, repeatable and auditable way.

## Infrastructure as Code

Define Azure resources using text files.

## Declarative Syntax

Declare how the resources should be and Azure Resource Manager “makes it so”.

## JSON

ARM Templates are JSON format text files. Edit them in Visual Studio Code (or other text editors). Version control them.

## Meta-Language

Contains some programming language constructs such as functions and loops.

```
1   {
2     "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTe
3     "contentVersion": "1.0.0.0",
4     "parameters": {…
45    },
46    "variables": {
47      "webSiteName": "[concat('webSite', uniqueString(resourceGroup().id))]"
48    },
49    "resources": [
50      {
51        "type": "Microsoft.Web/serverfarms", // dependency for a web site
52        "apiVersion": "2015-08-01",
53        "name": "[parameters('hostingPlanName')]",
54        "location": "[resourceGroup().location]",
55        "tags": {
56          "environment": "Production" // change this if not prod
57        },
58        "sku": {
59          "name": "[parameters('skuName')]",
60          "capacity": "[parameters('skuCapacity')]"
61        },
62        "properties": {
63          "name": "[parameters('hostingPlanName')]"
64        }
65      },
66      {
```

https://docs.microsoft.com/en-us/azure/templates/

# Features of ARM Templates

## Parameterized

Parameters can be used to configure resource attributes at deployment time. Allows generalization and re-use of ARM Templates.

## Modular

Templates can be broken into smaller, re-usable components and **linked** together at deployment time by using **Deployment** resources.

Templates can also be **nested** inside other templates.

## Idempotent

Resources will only be changed or created if they have drifted out of state or need to be updated or created.

3 deployment modes: Complete, Increment, Validate

## Testable

Templates can be validated prior to deployment.

## Version Control

Using ARM Templates with version control allows your infrastructure to be reviewable, traceable and auditable.

https://docs.microsoft.com/en-us/azure/azure-resource-manager/template-best-practices

# Imperative vs Declarative

## Imperative Code (HOW)

***Defines specific commands required to reach a desired state***, and the order and understanding of each command is critical.

## Declarative (WHAT)

***Defines desired state,*** *and the details about how that is achieved are largely irrelevant.*

***Problem statement:*** *we need a new environment for a financial application.*

```
New-AzVirtualNetwork $financeNetwork
New-AzAvailableSet $appAS
New-AzVM $appVM01
New-AzVM $appVM02
New-AzAvailableSet $sqlAS
New-AzVM $sqlVM01
New-AzVM $sqlVM02
New-AzWebApp $webApp
```

***financeAppTemplate.json***

## Author

Describe environments

### ARM Tools - VS Code Extension

Simplified authoring experience with the new VS Code extension for ARM Templates

## Validate

Impact assessment & best practice checks

### What-IF & ARM TTK & AzSK

Pre-deployment impact assessment and drift detection

Static analysis and testing

## Deploy

Staging & Orchestrated deployments

### Template Specs & Deployment Scripts

Easy staging and sharing of templates and artifacts

Complete the last mile of your deployments

## Manage

Manage your environment

### 'Deployment Stacks'

Easily update, rollback and delete environments

**Provision and manage the lifecycle of resources in a SIMPLE declarative way**

# ARM Template Schema

## Whole structure

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

# ARM Template Schema

## Parameters

```
{
  "schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "",
  "apiProfile": "",
  "parameters": {
      "environment": {
         "allowedValues": ["dev", "test", "stg", "prod"],
         "type": "string"
      }
  },
  "variables": {},
  "functions": [  ],
  "resources": [  ],
  "outputs": {  }
}
```

- Parameters are values to be passed to the template at time of deployment
- Examples: Environment, size of resources, resource group name, …

# ARM Template Schema

## Variables

```
{
  "schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "",
  "apiProfile": "",
  "parameters": {},
  "variables": {
    "resourceGroupName": "[concat('rg-’,
            parameters('rgName'), ’-’,
            parameters('environment'))]"
   },
  "functions": [  ],
  "resources": [  ],
  "outputs": {  }
}
```

- Variables are values that are hard coded to be reused in the template
- Examples: options, naming conventions,

# ARM Template Schema

## Functions

```
{"schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
 "contentVersion": "",
 "functions": [
 {
   "namespace": "contoso",
   "members": {
     "uniqueName": {
       "parameters": [
         {
           "name": "namePrefix",
           "type": "string"
         }
       ],
       "output": {
         "type": "string",
         "value": "[concat(toLower(parameters('namePrefix')), uniqueString(resourceGroup().id))]"
       }
     }}
}],
```

- Use-defined function defines complicated expressions that you don't want to repeat throughout your template.
- Examples: options, naming conventions,

# ARM Template Schema

## Resources

```
{"schema": "https://schema.management.azure.com/schemas/2019-04-01/d
  "contentVersion": "",
  "resources": [
  {
    "name": "contosoStorage",
    "type": "Microsoft.Storage/storageAccounts",
    "apiVersion": "2015-06-15",
    "location": "[resourceGroup().location]",
    "properties": [
      "accountType": "RA_GRS"
    ]
  }],
  "functions": [  ],
  “parameters”, {},
  “variables”: {},
  "outputs": {  }
}
```

| Element name | Required |
|---|---|
| condition | No |
| apiVersion | Yes |
| type | Yes |
| name | Yes |
| location | Varies |
| tags | No |
| comments | No |
| copy | No |
| dependsOn | No |
| properties | No |
| sku | No |
| kind | No |
| plan | No |
| resources | No |

- Resource you want to deploy.
- Examples: VM, storage, DB, Vnet, …

# ARM Template Schema

## ApiProfile

```
{"schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "",
  "apiProfile": "2018-03-01-hybrid",
  "funcrions": [],
  "resources": [  ],
  “parameters”, {},
  “variables”: {},
  “apiVerions": ""
}
```

- Use this value to avoid having to specify API versions for each resource in the template.
- Available API profile version: https://github.com/Azure/azure-rest-api-specs/tree/master/profile .

# ARM Template Schema

## Output

```
{"schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "",
  "outputs": {
      "resourceID": {
        "condition": "[equals(parameters('publicIpNewOrExisting'), 'new')]",
        "type": "string",
        "value": "[resourceId('Microsoft.Network/publicIPAddresses', parameters('publicIPAddresses_name'))]"
      }
  },
  "funcrions": [],
  "resources": [  ],
  “parameters”, {},
  “variables”: {},
  “apiVerions": ""
}
```

- Used to exchange data between resources in templates.
- Example 1: create a public IP and pass that resource ID to a public load balancers
- Example 2: create a naming template to pass resource names to deployment.

# Demo

- Deploy template
- Export template
- Visualize deployed resources on Azure
- Azure ARM alternative Azure/bicep
- Project Bicep Demo at Ignite 2020 by Mark Russinovich | Azure DevOps Blog (microsoft.com)

slido

# Are you applying infrastructure as code?

ⓘ Start presenting to display the poll results on this slide.

slido

# What challenges you have?

ⓘ Start presenting to display the poll results on this slide.

slido

# Which tool set you are using?

ⓘ Start presenting to display the poll results on this slide.

# Infrastructure Code change process automation

# Introducing Azure DevOps

## Azure Boards

Deliver value to your users faster using proven agile tools to plan, track, and discuss work across your teams.

## Azure Pipelines

Build, test, and deploy with CI/CD that works with any language, platform, and cloud. Connect to GitHub or any other Git provider and deploy continuously.

## Azure Repos

Get unlimited, cloud-hosted private Git repos and collaborate to build better code with pull requests and advanced file management.

## Azure Test Plans

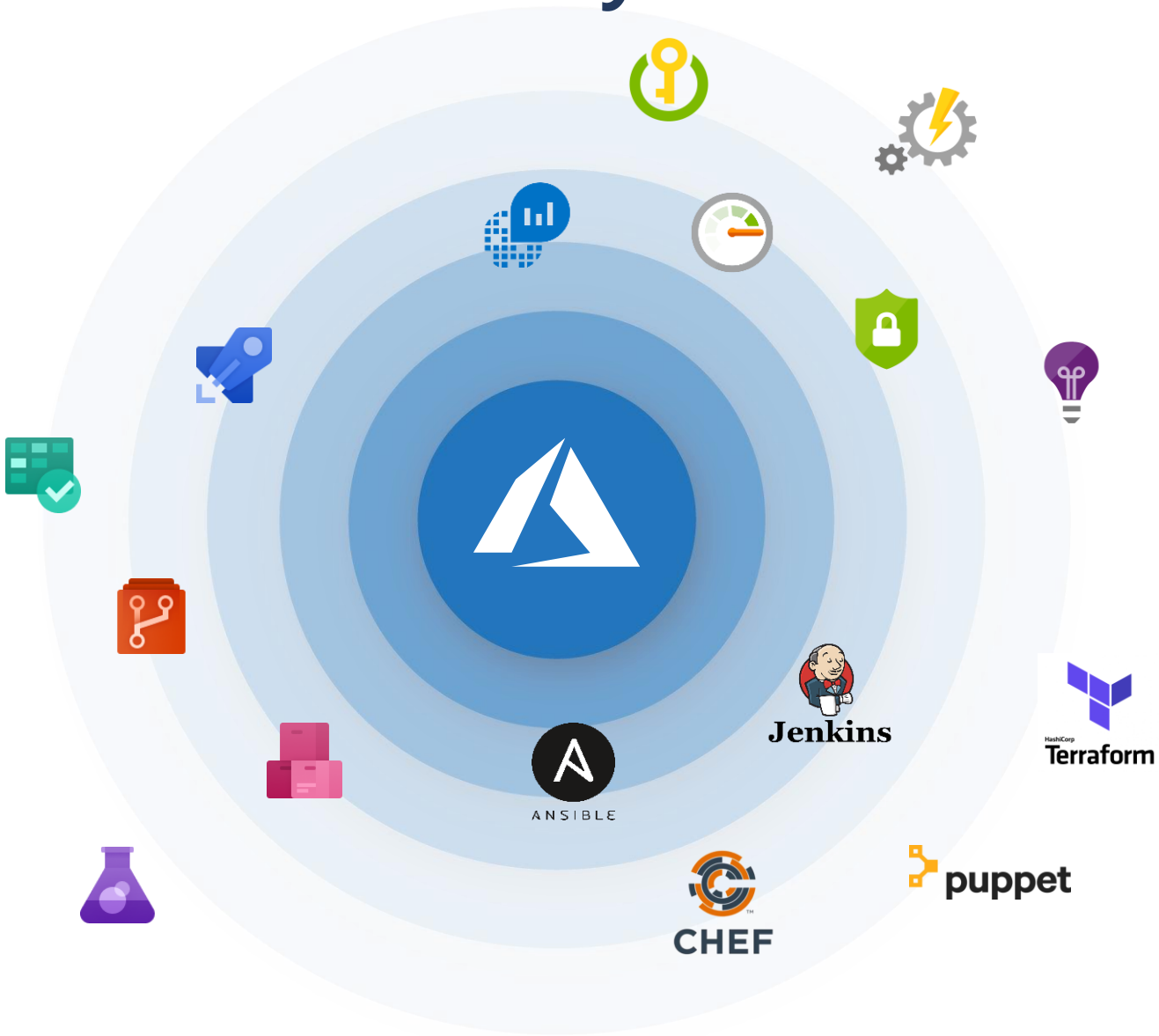
Test and ship with confidence using manual and exploratory testing tools.

## Azure Artifacts

Create, host, and share packages with your team, and add artifacts to your CI/CD pipelines with a single click.

https://azure.com/devops

# Broadening the Azure Ecosystem



# Azure Pipelines

Cloud-hosted pipelines for Linux, Windows and macOS, with unlimited minutes for open source

## Any language, any platform, any cloud

Build, test, and deploy Node.js, Python, Java, PHP, Ruby, C/C++, .NET, Android, and iOS apps. Run in parallel on Linux, macOS, and Windows. Deploy to Azure, AWS, GCP or on-premises

## Extensible

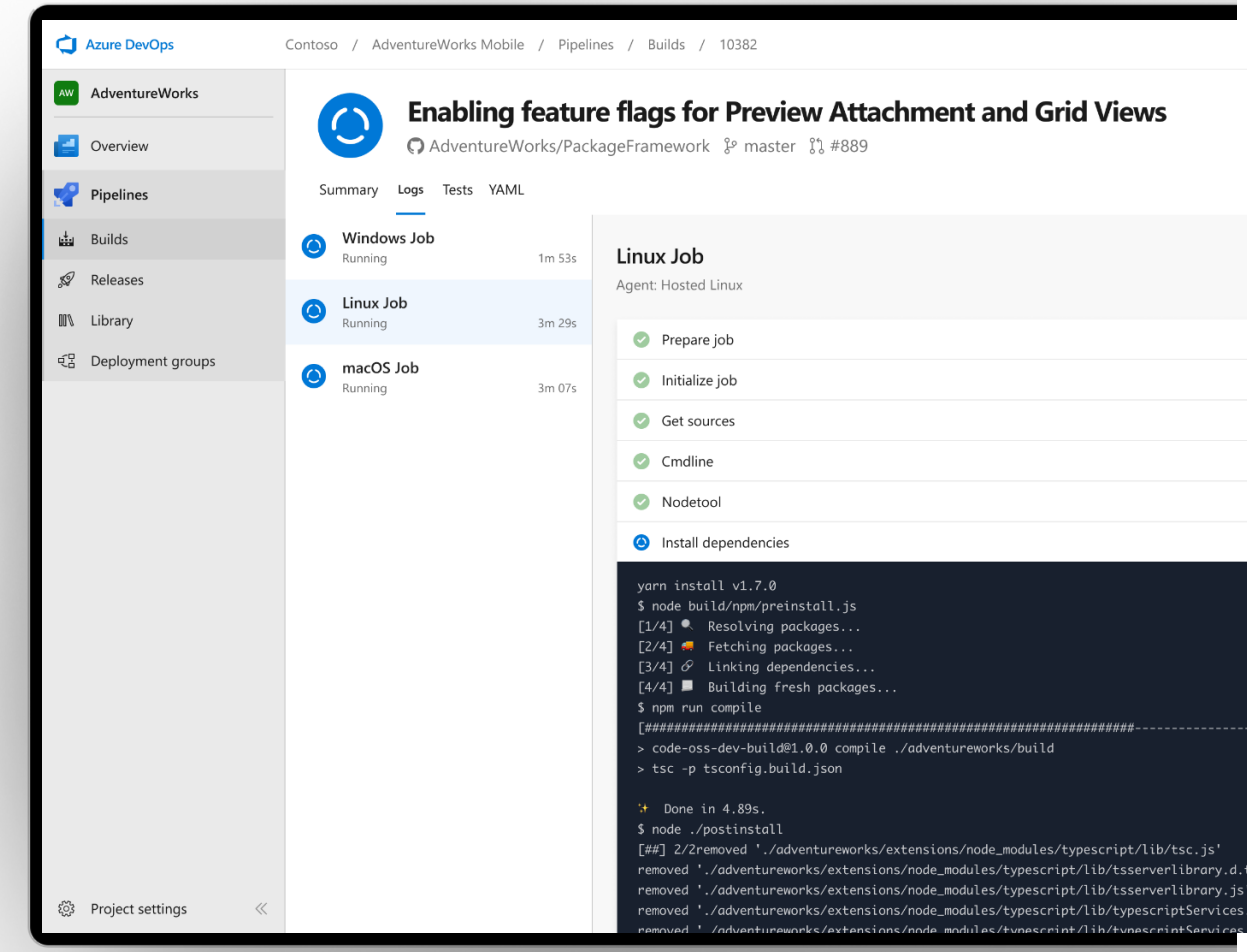
Explore and implement a wide range of community-built build, test, and deployment tasks, along with hundreds of extensions from Slack to SonarCloud. Support for YAML, reporting and more

## Containers and Kubernetes

Easily build and push images to container registries like Docker Hub and Azure Container Registry. Deploy containers to individual hosts or Kubernetes.

## Best-in-class for open source

Ensure fast continuous integration/continuous delivery (CI/CD) pipelines for every open source project. Get unlimited build minutes for all open source projects with up to 10 free parallel jobs across Linux, macOS and Windows




https://azure.com/pipelines

# Azure Pipelines

Free **unlimited** build minutes for public projects

Up to 10 free parallel jobs across Windows, Linux and macOS

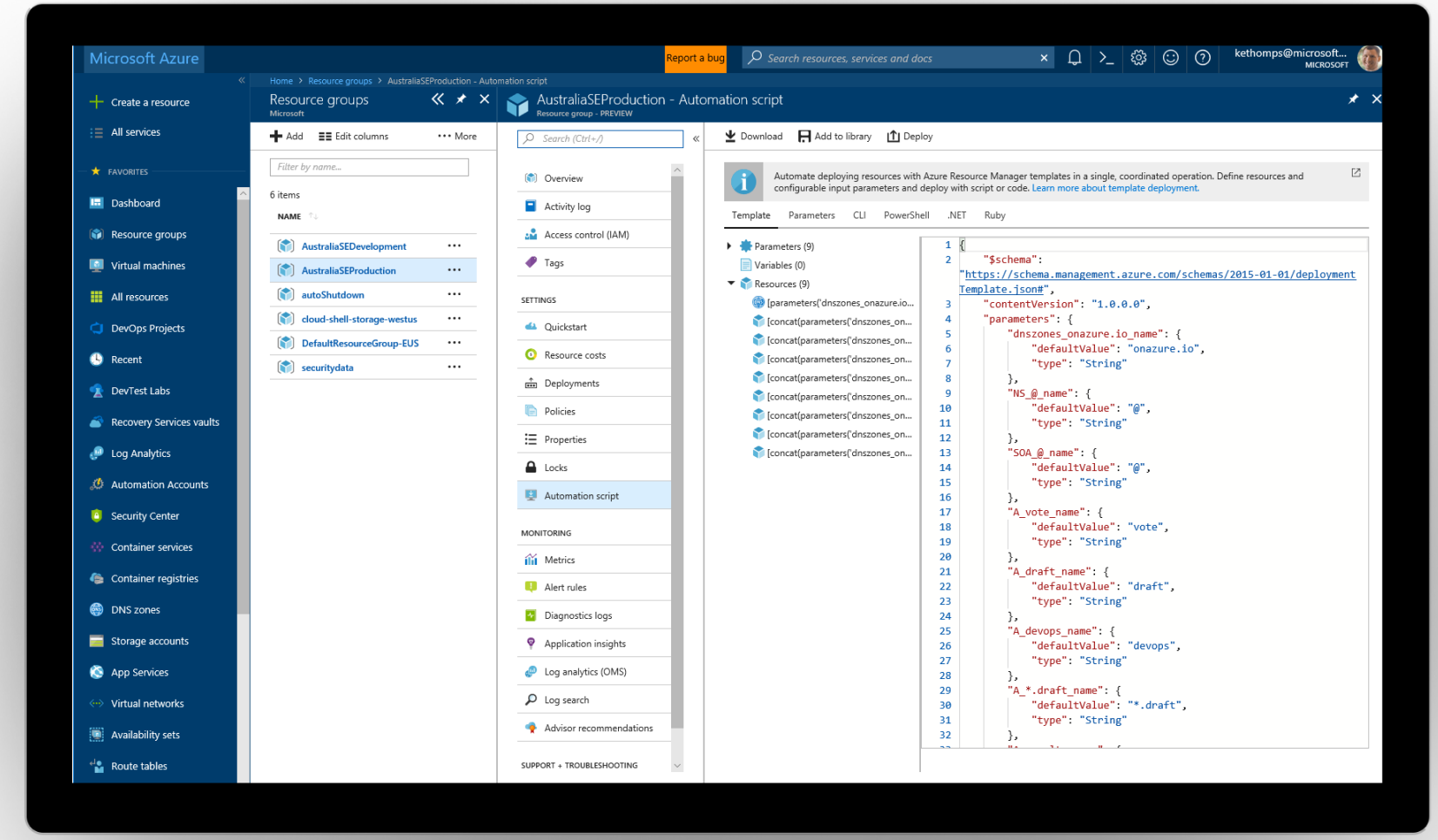
https://azure.com/pipelines

Microsoft ❤ Open Source

# Infrastructure and Configuration as Code

## Azure Resource Manager, Automation & 3rd Party Integrations

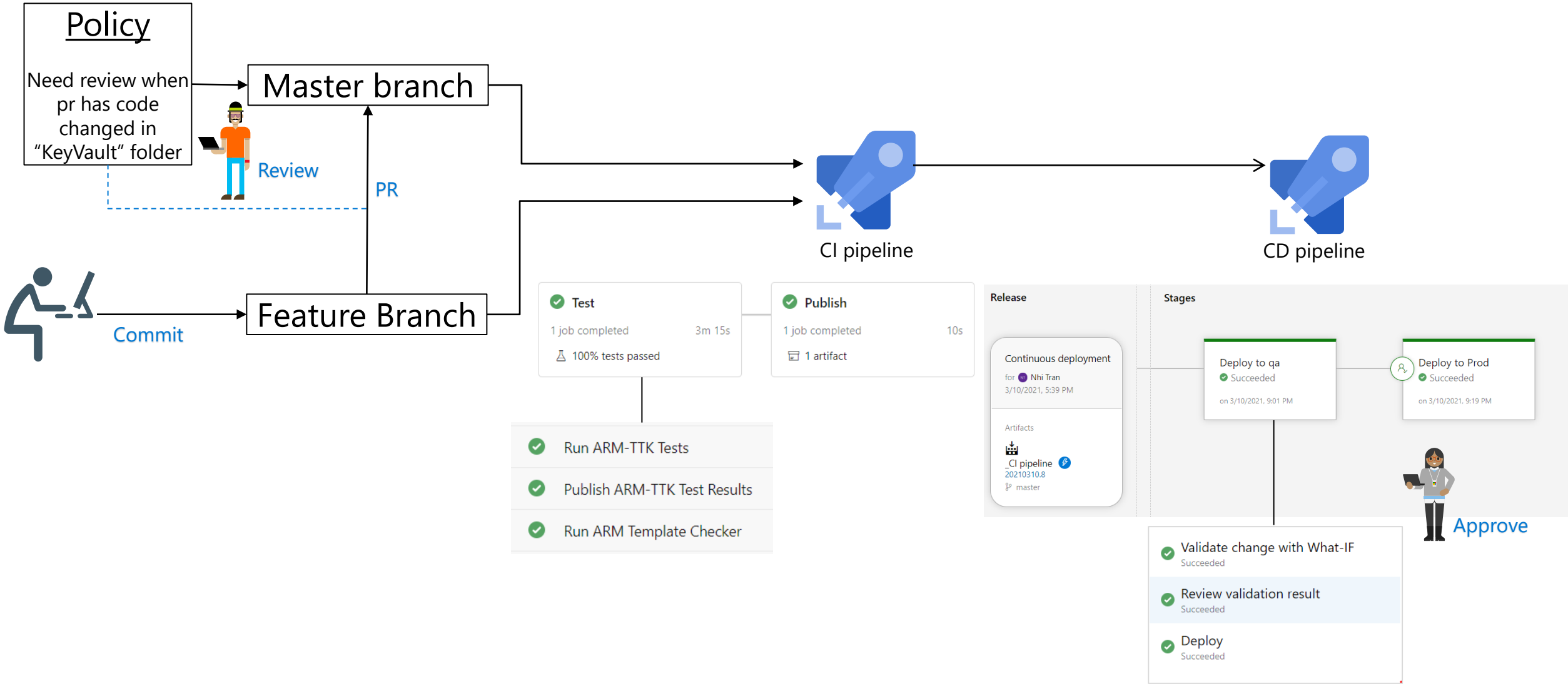
- Infrastructure as Code, built-in
- Azure Config & Automation
- Support for 3rd party and OSS tooling such as Terraform, Ansible, Chef, Puppet & SaltStack




TERRAFORM ANSIBLE CHEF puppet SALTSTACK

# Demo

- Automate IaC change process with DevOps
- Test your template
- Security validation for your template
- What-if impact assessment


Policy
Need review when pr has code changed in "KeyVault" folder
Master branch
Review
PR
Commit
Feature Branch
CI pipeline
CD pipeline
Test
1 job completed
3m 15s
100% tests passed
Publish
1 job completed
10s
1 artifact
Run ARM-TTK Tests
Publish ARM-TTK Test Results
Run ARM Template Checker
Release
Stages
Continuous deployment
for Nhi Tran
3/10/2021, 5:39 PM
Artifacts
_CI pipeline
20210310.8
master
Deploy to qa
Succeeded
on 3/10/2021, 9:01 PM
Deploy to Prod
Succeeded
on 3/10/2021, 9:19 PM
Approve
Validate change with What-IF
Succeeded
Review validation result
Succeeded
Deploy
Succeeded

# Azure Resource Manager Template Toolkit (arm-ttk)

The code in this repository can be used for analyzing and testing Azure Resource Manager Templates. The tests will check a template or set of templates for coding best practices. There are some checks for simple syntactical errors but the intent is not to re-implement tests or checks that are provided by the platform (e.g. the /validate api).

- Installing
- Run test

Test-AzTemplate -TemplatePath $TemplateFolder

Test-AzTemplate -TemplatePath $TemplateFolder -File cdn.json

Test-AzTemplate -TemplatePath $TemplateFolder -Test "Resources Should Have Location“

- Customizable with your own test
- Result format

```
    [+] adminUsername Should Not Be A Literal (11 ms)

    [+] apiVersions Should Be Recent (10 ms)
    [+] artifacts parameter (6 ms)
    [+] DeploymentTemplate Schema Is Correct (5 ms)
    [+] IDs Should Be Derived From ResourceIDs (7 ms)
    [-] Location Should Not Be Hardcoded (5 ms)
        azuredeploy.json must use the location parameter, not resourceGroup().location
lue in the main template)
```

## Default test cases

1. Use correct schema
2. Parameters must exist
3. Declared parameters must be used
4. Secure parameters can't have hardcoded default
5. Environment URLs can't be hardcoded
6. Location uses parameter
7. Resources should have location
8. VM size uses parameter
9. Min and max values are numbers
10. Artifacts parameter defined correctly
11. Declared variables must be used
12. Dynamic variable should not use concat
13. Use recent API version
14. Use hardcoded API version
15. Properties can't be empty
16. Use Resource ID functions
17. ResourceId function has correct parameters
18. dependsOn best practices
19. Nested or linked deployments can't use debug
20. Admin user names can't be literal value
21. Use latest VM image
22. Use stable VM images
23. Don't use ManagedIdentity extension
24. Outputs can't include secrets
25. Next steps

# Azure arm-ttk plugin for DevOps

https://marketplace.visualstudio.com/items?itemName=Sam-Cogan.ARMTTKExtension

Visual Studio | Marketplace

Nhi Tran (

Azure DevOps > Azure Pipelines > Run ARM TTK Tests

## Run ARM TTK Tests

Sam Cogan | 691 installs | ★★★★★ (4) | Free

Run Azure Resource Manager Template Test Kit tests as part of your build or release process

Get it free

Overview Q & A Rating & Review

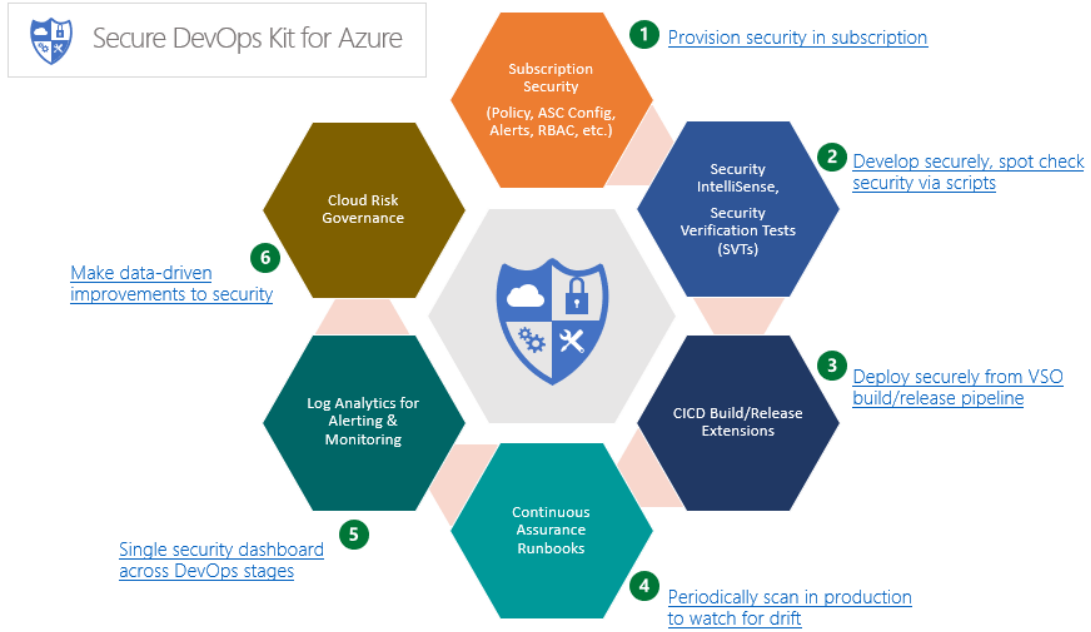
### arm-ttk-extension

# Secure DevOps Kit for Azure (AzSK)

## Overview

The "Secure DevOps Kit for Azure" (will be referred to as 'AzSK' henceforth) is a collection of scripts, tools, extensions, automations, etc. that caters to the end to end Azure subscription and resource security needs for dev ops teams using extensive automation and smoothly integrating security into native dev ops workflows helping accomplish secure dev ops with these 6 focus areas:

1. `Secure the subscription`: A secure cloud subscription provides a core foundation upon which subsequent development and deployment activities can be conducted. An engineering team should have the capabilities to deploy and configure security in the subscription including elements such as alerts, ARM policies, RBAC, Security Center policies, JEA, Resource Locks, etc. Likewise, it should be possible to check that all settings are in conformance to a secure baseline.
2. `Enable secure development`: During the coding and early development stages, developers should have the ability to write secure code and to test the secure configuration of their cloud applications. Just like *build verification tests* (BVTs), we introduce the concept of *security verification tests* (SVTs) which can check for security of various resource types in Azure.
3. `Integrate security into CICD`: Test automation is a core tenet of devops. We emphasize this by providing the ability to run SVTs as part of the VSTS CICD pipeline. These SVTs can be used to ensure that the target subscription used to deploy a cloud application and the Azure resources the application is built upon are all setup in a secure manner.
4. `Continuous Assurance`: In the constantly changing dev ops environment, it is important to move away from the mindset of security being a milestone. We have to treat security as a *continuously varying state* of a system. This is made possible through capabilities that enable *continuous assurance* using a combination of automation runbooks, schedules, etc.
5. `Alerting & Monitoring`: Visibility of security status is important for individual application teams and also for central enterprise teams. We provide solutions that cater to the needs of both. Moreover, the solution spans across all stages of dev ops in effect bridging the gap between the *dev* team and the *ops* team from a security standpoint through the single, integrated views it generates.
6. `Cloud Risk Governance`: Lastly, underlying all activities in the kit is a telemetry framework that generates events capturing usage, adoption, evaluation results, etc. This allows us to make measured improvements to security targeting areas of high risk and maximum usage before others.




[(Click me) Secure DevOps Kit for Azure (azsk.azurewebsites.net)](azsk.azurewebsites.net)

# Azure AzSK plugin for DevOps

https://marketplace.visualstudio.com/items?itemName=azsdktm.AzSDK-task

Visual Studio | Marketplace

Nhi Tran (nhitran@

Azure DevOps > Azure Pipelines > Secure DevOps Kit (AzSK) CICD Extensions for Azure

## Secure DevOps Kit (AzSK) CICD Extensions for Azure

Microsoft | 5,907 installs | ★★★★★ (7) | Free

Collection of extensions that empower DevOps teams to build and deploy applications on Azure with security integrated at every step.

Get it free

Overview Q & A Rating & Review

### Secure DevOps Kit for Azure (AzSK) - CICD VSTS extension

# ARM Template What-if

Display the changes that will be made to a resource group when the ARM template is deployed.

### Validate Changes Before Deployment

Protect against harmful or unexpected changes before deploying to production.

### OSS PowerShell & AZ CLI

Supports Linux, Windows and MacOS.
Supported version: Az **4.2 later**

### Integrate into DevOps Processes

GitHub Actions or Azure DevOps Pipelines. Branch Policy, CI, Deployment.

```
dascottr@MININT-LL1TSG6  ~\source\AzureDevOps\dscottraynsford\Demonstrations\Ironcla
> New-AzResourceGroupDeployment `
>>     -TemplateFile '.\src\infrastructure\all\azuredeploy.json' `
>>     -TemplateParameterObject @{ Location = 'EastUS' } `
>>     -ResourceGroupName 'dsr-ironclad-rg' `
>>     -WhatIf
```

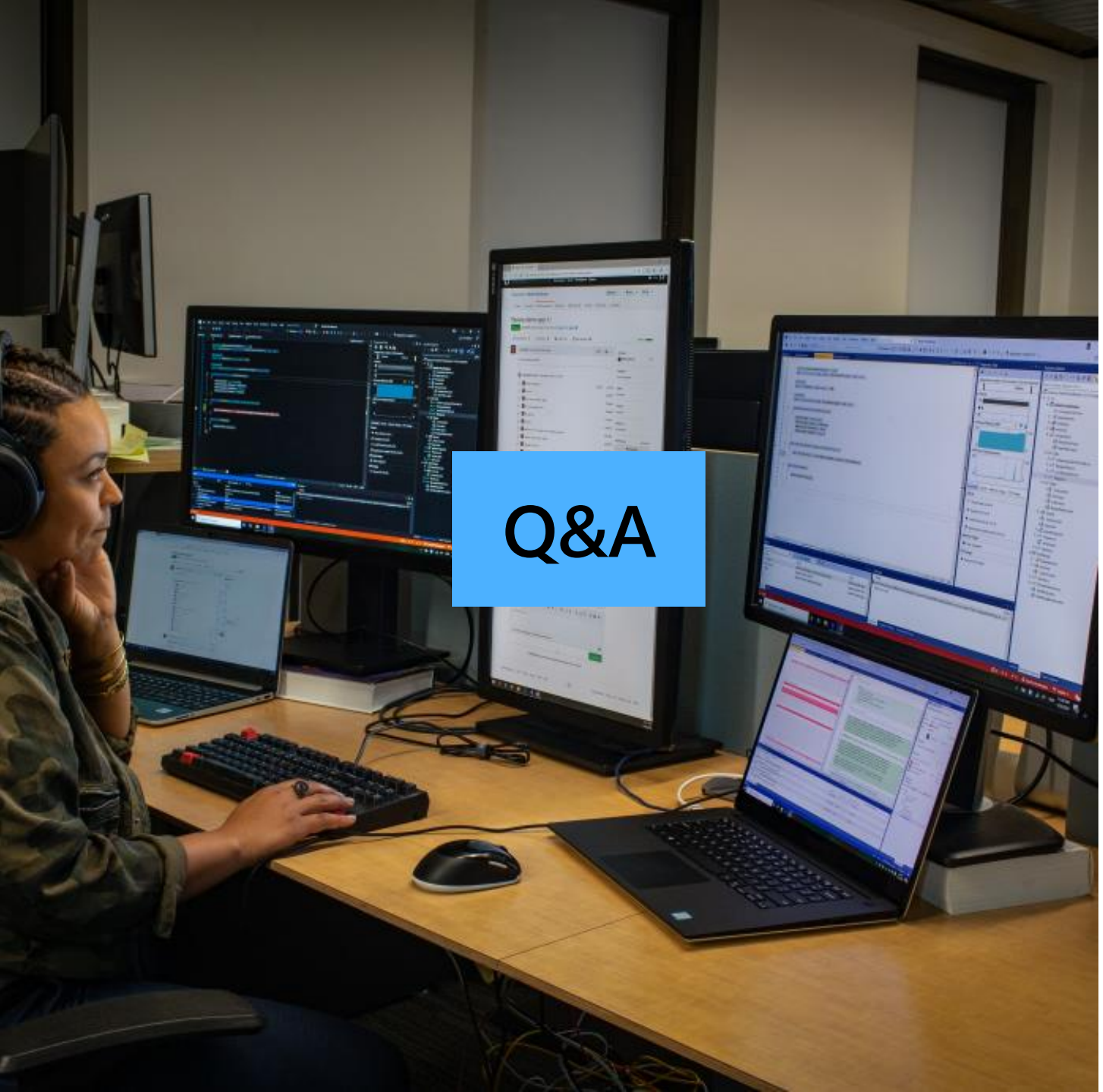
```
Resource and property changes are indicated with these symbols:
  - Delete
  + Create
  ~ Modify

The deployment will update the following scope:

Scope: /subscriptions/./resourceGroups/ExampleGroup

  ~ Microsoft.Network/virtualNetworks/vnet-001 [2018-10-01]
    - tags.Owner: "Team A"
    ~ properties.addressSpace.addressPrefixes: [
      - 0: "10.0.0.0/16"
      + 0: "10.0.0.0/15"
      ]
    ~ properties.subnets: [
      - 0:

          name:                       "subnet001"
          properties.addressPrefix: "10.0.0.0/24"

      ]

Resource changes: 1 to modify.
```

```
false positive predicti
issue here: https://aka

rceGroups/dsr-ironclad-r

6-01]

72ad9153-ecab-48c9-8a7a-

ety"

ge/storageAccounts"
```

# Kahoot

Contact us at: saaslab@microsoft.com
Appreciate your feedback at : https://aka.ms/saaslabfeedback4

Feel free to use program channel to post your comment and question.




# Your feedback is important

Contact us at: saaslab@microsoft.com

Please help us improve this program by completing this short feedback form.

https://aka.ms/saaslabfeedback4

# Additional resources – IaC validation

**Repository of the demo**

Github repository: microsoft/saaslab: SaaS-ification resources for ISVs (github.com)

**ARM alternative**

Github repository: Azure/bicep

Bicep playground: Bicep Playground 0.2.212 (windows.net)

**ARM-TTK**

Documentation: ARM template test toolkit - Azure Resource Manager | Microsoft Docs

GitHub repository: Azure/arm-ttk: Azure Resource Manager Template Toolkit (github.com)

Azure DevOps ARM-TTK Task: Run ARM TTK Tests - Visual Studio Marketplace (visualstudio.com)

**Secure DevOps Kit for Azure**

Documentation: Secure DevOps Kit for Azure (azsk.azurewebsites.net)

AzSK ARM Template Checker: Security Verification Tests (SVTs) | AzSK website

Secure DevOps Kit (AzSK) CICD Extensions for Azure: Secure DevOps Kit (AzSK) CICD Extensions for Azure

**ARM Template What-If**

What-If: Template deployment what-if (Preview) - Azure Resource Manager | Microsoft Docs

# Additional resources - Devops

**Azure Resource Template Manager**
https://docs.microsoft.com/en-us/azure/azure-resource-manager/ ‘

**Azure Resource Manager Template Reference**
https://docs.microsoft.com/en-us/azure/templates/

**ARM Template GitHub Quickstart**
https://github.com/azure/azure-quickstart-templates

**Configuring CI/CD Pipelines as Code with YAML in Azure DevOps**
https://www.azuredevopslabs.com/labs/azuredevops/yaml

**DevOps with Github learning journey**
https://partner.microsoft.com/en-US/training/assets/collection/devops-with-github-learning-journe

# Addition Resources – Source Code control

## Branch Policies

Azure DevOps: Protect your Git branches with policies - Azure Repos | Microsoft Docs
GitHub: About protected branches - GitHub Docs

## Code Review

Azure DevOps: Review and merge code with pull requests - Azure Repos | Microsoft Docs
GitHub: Features · Code review (github.com)

## Pipelines

Azure DevOps: Azure Pipelines documentation | Microsoft Docs
GitHub: GitHub Actions Documentation - GitHub Docs

## Environments

Azure DevOps: Environment - Azure Pipelines | Microsoft Docs
GitHub: Not Available.

# Additional resource - Template

| | |
|---|---|
| Azure Templates Quick Starts | https://github.com/Azure/azure-quickstart-templates |
| Best Practices | https://github.com/Azure/azure-quickstart-templates/blob/master/1-CONTRIBUTION-GUIDE/best-practices.md |
| Template Validation Tool | https://github.com/Azure/azure-quickstart-templates/tree/master/test/template-validation-tests |
| Template Deployment Scripts | PowerShell – https://github.com/Azure/azure-quickstart-templates/blob/master/Deploy-AzureResourceGroup.ps1<br>Bash – https://github.com/Azure/azure-quickstart-templates/blob/master/az-group-deploy.sh |
| UI Testing SideLoad Scripts: | PowerShell – https://github.com/Azure/azure-quickstart-templates/blob/master/SideLoad-CreateUIDefinition.ps1<br>Bash – https://github.com/Azure/azure-quickstart-templates/blob/master/sideload-createuidef.sh |
| Template Reference Docs | https://docs.microsoft.com/en-us/azure/templates/ |
| CreateUIDefinition Docs | https://docs.microsoft.com/en-us/azure/managed-applications/create-uidefinition-functions |
| Template Language Expressions | https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-template-functions |
| Azure PowerShell | https://docs.microsoft.com/en-us/powershell/azure/install-azurerm-ps?view=azurermps-5.7.0 |
| Azure CLI | https://docs.microsoft.com/en-us/cli/azure/?view=azure-cli-latest |
| Visual Studio Code Extension | https://marketplace.visualstudio.com/items?itemName=msazurermtools.azurerm-vscode-tools |