STREAMLINING DATABRICKS

CI/CD your Notebooks with DevOps Pipelines and orchestrate via Azure Data Factory

TechTacoFriday

Technology spiced with a taco flavor mix

“A discussion on whether use Azure Data Factory and Databricks together is like discussing whether to use a hammer and a screwdriver for any purpose: they are simply different tools, each with unique capabilities and you will achieve a superior result by using the strengths of both. So, let's take advantage of all orchestrating capabilities from Azure Data Factory to support any big data processing, analytics, and machine learning workloads from Databricks!!“

Hector Sven, Linkedin post March 2024

Workshop's Agenda

Provision your Workspace infrastructure with BICEP

Manage Users with Azure's Entra ID

Deploying Notebooks across environments with DevOps YAML Pipelines

Orchestrate your Notebooks via Azure Data Factory

About me...

Hector Sven

- ✓ From Mexico, living in Norway since Sept 2018
- ✓ Manager, Data Engineering @Avanade
- ✓ 20+ years in IT industry
- ✓ 7 associate certifications on Azure data-related technologies and a DevOps expert
- ✓ My blog >>>>>>>>>>>>>>>>>>
- ✓ Contact me
  - ❑ hector.lopez@avanade.com
  - ❑ svenchio@techtacofriday.com
  - ❑ www.linkedin.com/in/svenchio/

TechTacoFriday

Technology spiced with a taco flavor mix

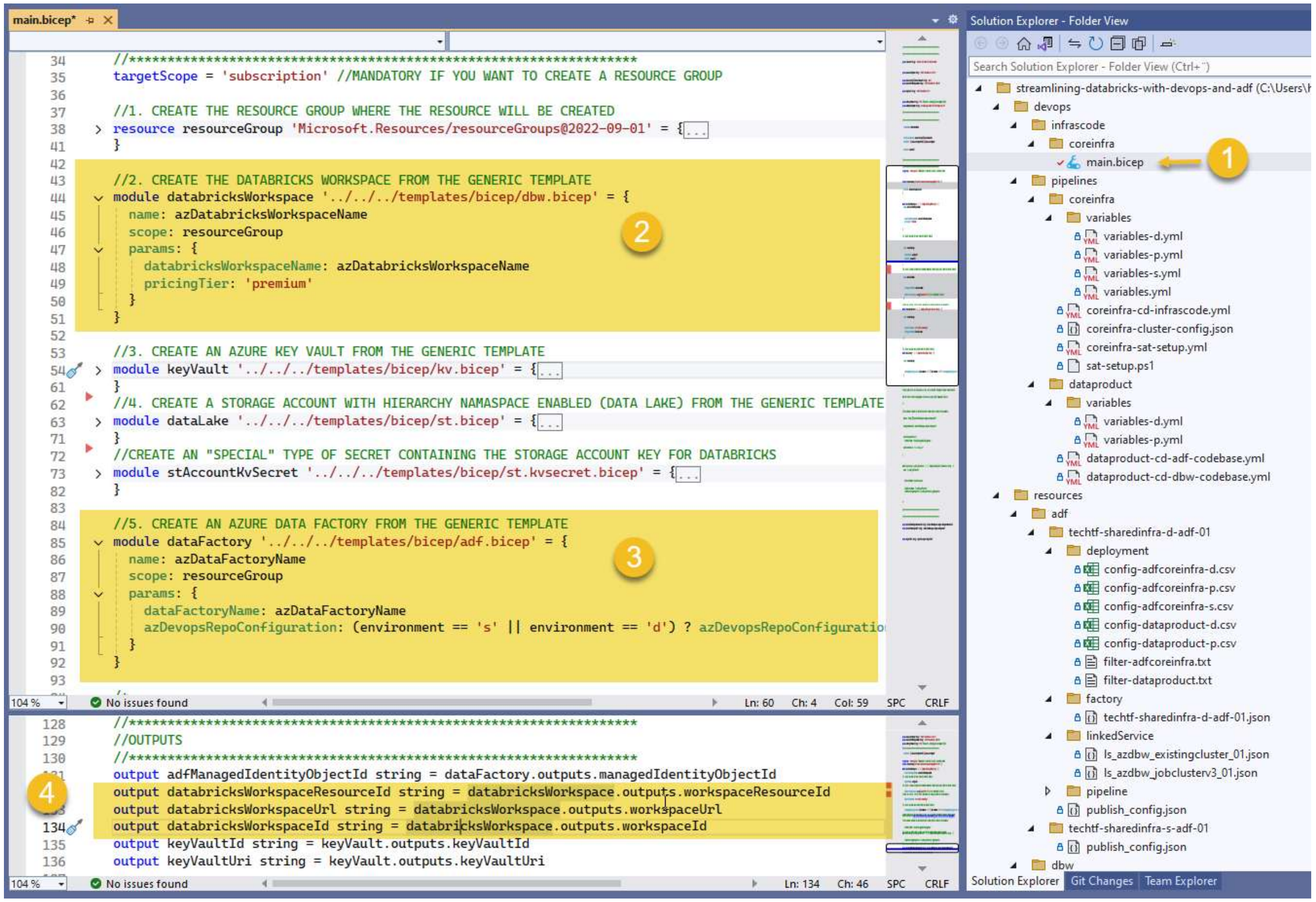
SVENCHIO

PROVISIONING DATABRICKS WORKSPACE INFRASTRUCTURE WITH BICEP

TechTacoFriday

Technology spiced with a taco flavor mix

A few pointers on BICEP


main.bicep*
Solution Explorer - Folder View
Search Solution Explorer - Folder View (Ctrl+")
1
2
3
4
Solution Explorer
Git Changes
Team Explorer

```
//*********************************************************************
targetScope = 'subscription' //MANDATORY IF YOU WANT TO CREATE A RESOURCE GROUP

//1. CREATE THE RESOURCE GROUP WHERE THE RESOURCE WILL BE CREATED
resource resourceGroup 'Microsoft.Resources/resourceGroups@2022-09-01' = {...
}

//2. CREATE THE DATABRICKS WORKSPACE FROM THE GENERIC TEMPLATE
module databricksWorkspace '../../../templates/bicep/dbw.bicep' = {
  name: azDatabricksWorkspaceName
  scope: resourceGroup
  params: {
    databricksWorkspaceName: azDatabricksWorkspaceName
    pricingTier: 'premium'
  }
}

//3. CREATE AN AZURE KEY VAULT FROM THE GENERIC TEMPLATE
module keyVault '../../../templates/bicep/kv.bicep' = {...
}
//4. CREATE A STORAGE ACCOUNT WITH HIERARCHY NAMASPACE ENABLED (DATA LAKE) FROM THE GENERIC TEMPLATE
module dataLake '../../../templates/bicep/st.bicep' = {...
}
//CREATE AN "SPECIAL" TYPE OF SECRET CONTAINING THE STORAGE ACCOUNT KEY FOR DATABRICKS
module stAccountKvSecret '../../../templates/bicep/st.kvsecret.bicep' = {...
}

//5. CREATE AN AZURE DATA FACTORY FROM THE GENERIC TEMPLATE
module dataFactory '../../../templates/bicep/adf.bicep' = {
  name: azDataFactoryName
  scope: resourceGroup
  params: {
    dataFactoryName: azDataFactoryName
    azDevopsRepoConfiguration: (environment == 's' || environment == 'd') ? azDevopsRepoConfiguratio
  }
}
```

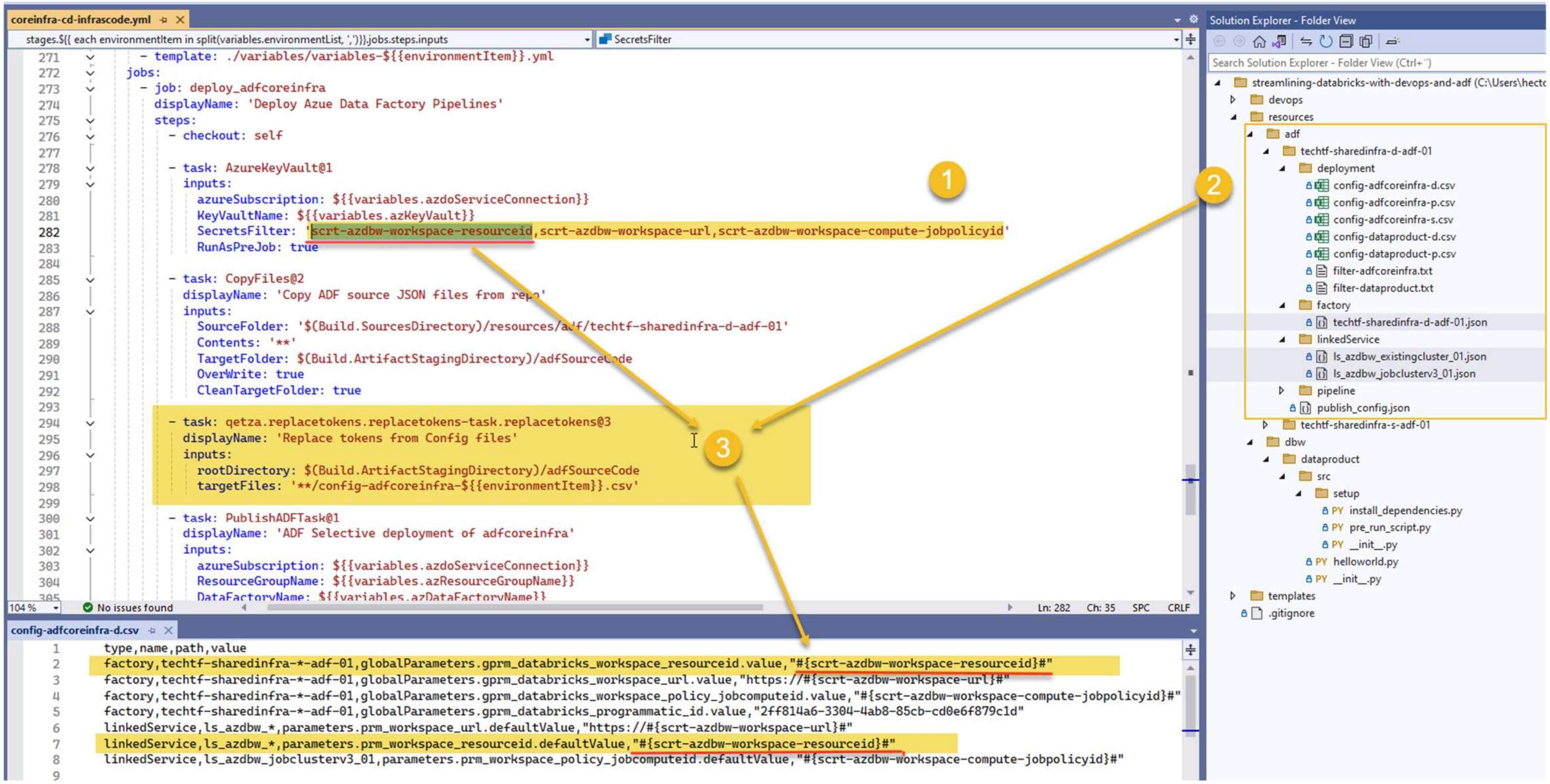
104 % No issues found Ln: 60 Ch: 4 Col: 59 SPC CRLF

```
//*********************************************************************
//OUTPUTS
//*********************************************************************
output adfManagedIdentityObjectId string = dataFactory.outputs.managedIdentityObjectId
output databricksWorkspaceResourceId string = databricksWorkspace.outputs.workspaceResourceId
output databricksWorkspaceUrl string = databricksWorkspace.outputs.workspaceUrl
output databricksWorkspaceId string = databricksWorkspace.outputs.workspaceId
output keyVaultId string = keyVault.outputs.keyVaultId
output keyVaultUri string = keyVault.outputs.keyVaultUri
```

104 % No issues found Ln: 134 Ch: 46 SPC CRLF

- streamlining-databricks-with-devops-and-adf (C:\Users\
  - devops
    - infrascode
      - coreinfra
        - main.bicep
    - pipelines
      - coreinfra
        - variables
          - variables-d.yml
          - variables-p.yml
          - variables-s.yml
          - variables.yml
        - coreinfra-cd-infrascode.yml
        - coreinfra-cluster-config.json
        - coreinfra-sat-setup.yml
        - sat-setup.ps1
      - dataproduct
        - variables
          - variables-d.yml
          - variables-p.yml
        - dataproduct-cd-adf-codebase.yml
        - dataproduct-cd-dbw-codebase.yml
  - resources
    - adf
      - techtf-sharedinfra-d-adf-01
        - deployment
          - config-adfcoreinfra-d.csv
          - config-adfcoreinfra-p.csv
          - config-adfcoreinfra-s.csv
          - config-dataproduct-d.csv
          - config-dataproduct-p.csv
          - filter-adfcoreinfra.txt
          - filter-dataproduct.txt
        - factory
          - techtf-sharedinfra-d-adf-01.json
        - linkedService
          - ls_azdbw_existingcluster_01.json
          - ls_azdbw_jobclusterv3_01.json
        - pipeline
        - publish_config.json
      - techtf-sharedinfra-s-adf-01
        - publish_config.json
    - dbw

How it works in a nutshell

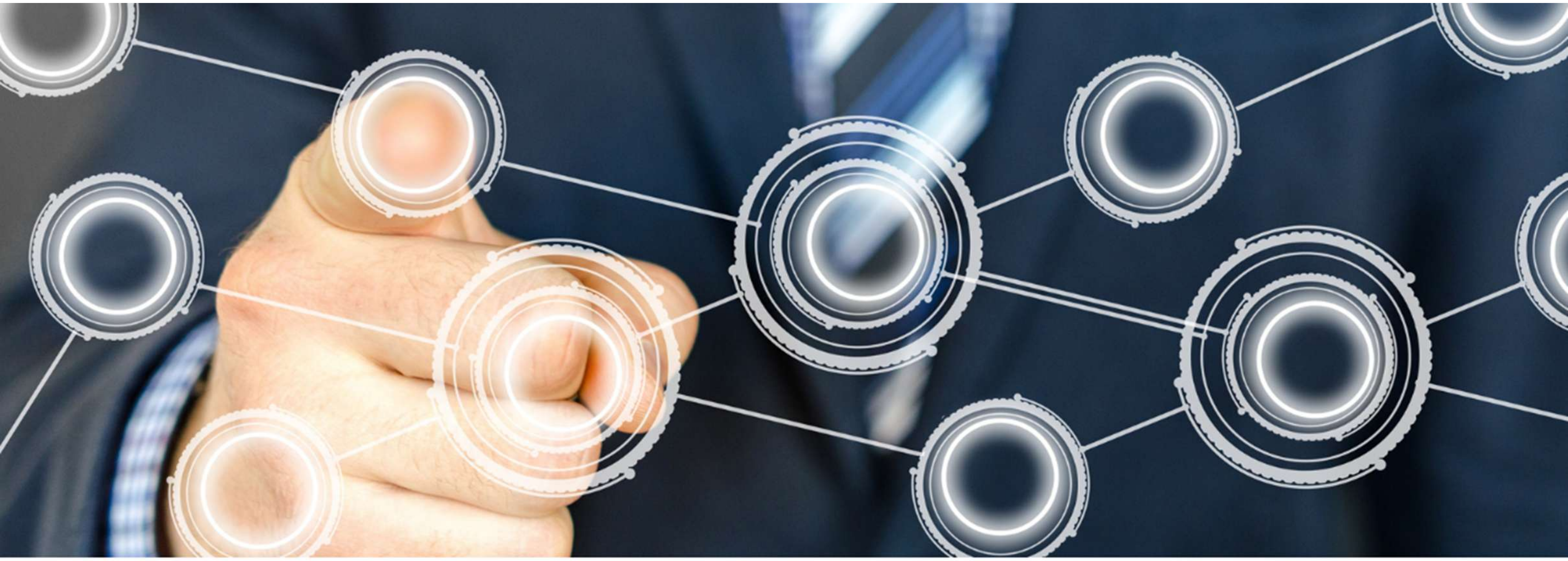


coreinfra-cd-infrascode.yml
stages.${{ each environmentItem in split(variables.environmentList, ',')}}.jobs.steps.inputs
SecretsFilter
1
2
3
104 %
No issues found
Ln: 282 Ch: 35 SPC CRLF
config-adfcoreinfra-d.csv
Solution Explorer - Folder View
Search Solution Explorer - Folder View (Ctrl+`)

```yaml
    - template: ./variables/variables-${{environmentItem}}.yml
  jobs:
    - job: deploy_adfcoreinfra
      displayName: 'Deploy Azue Data Factory Pipelines'
      steps:
        - checkout: self

        - task: AzureKeyVault@1
          inputs:
            azureSubscription: ${{variables.azdoServiceConnection}}
            KeyVaultName: ${{variables.azKeyVault}}
            SecretsFilter: 'scrt-azdbw-workspace-resourceid,scrt-azdbw-workspace-url,scrt-azdbw-workspace-compute-jobpolicyid'
            RunAsPreJob: true

        - task: CopyFiles@2
          displayName: 'Copy ADF source JSON files from repo'
          inputs:
            SourceFolder: '$(Build.SourcesDirectory)/resources/adf/techtf-sharedinfra-d-adf-01'
            Contents: '**'
            TargetFolder: $(Build.ArtifactStagingDirectory)/adfSourceCode
            OverWrite: true
            CleanTargetFolder: true

        - task: qetza.replacetokens.replacetokens-task.replacetokens@3
          displayName: 'Replace tokens from Config files'
          inputs:
            rootDirectory: $(Build.ArtifactStagingDirectory)/adfSourceCode
            targetFiles: '**/config-adfcoreinfra-${{environmentItem}}.csv'

        - task: PublishADFTask@1
          displayName: 'ADF Selective deployment of adfcoreinfra'
          inputs:
            azureSubscription: ${{variables.azdoServiceConnection}}
            ResourceGroupName: ${{variables.azResourceGroupName}}
            DataFactoryName: ${{variables.azDataFactoryName}}
```

```
type,name,path,value
factory,techtf-sharedinfra-*-adf-01,globalParameters.gprm_databricks_workspace_resourceid.value,"#{scrt-azdbw-workspace-resourceid}#"
factory,techtf-sharedinfra-*-adf-01,globalParameters.gprm_databricks_workspace_url.value,"https://#{scrt-azdbw-workspace-url}#"
factory,techtf-sharedinfra-*-adf-01,globalParameters.gprm_databricks_workspace_policy_jobcomputeid.value,"#{scrt-azdbw-workspace-compute-jobpolicyid}#"
factory,techtf-sharedinfra-*-adf-01,globalParameters.gprm_databricks_programmatic_id.value,"2ff814a6-3304-4ab8-85cb-cd0e6f879c1d"
linkedService,ls_azdbw_*,parameters.prm_workspace_url.defaultValue,"https://#{scrt-azdbw-workspace-url}#"
linkedService,ls_azdbw_*,parameters.prm_workspace_resourceid.defaultValue,"#{scrt-azdbw-workspace-resourceid}#"
linkedService,ls_azdbw_jobclusterv3_01,parameters.prm_workspace_policy_jobcomputeid.defaultValue,"#{scrt-azdbw-workspace-compute-jobpolicyid}#"
```

```
streamlining-databricks-with-devops-and-adf (C:\Users\hecto
  devops
  resources
    adf
      techtf-sharedinfra-d-adf-01
        deployment
          config-adfcoreinfra-d.csv
          config-adfcoreinfra-p.csv
          config-adfcoreinfra-s.csv
          config-dataproduct-d.csv
          config-dataproduct-p.csv
          filter-adfcoreinfra.txt
          filter-dataproduct.txt
        factory
          techtf-sharedinfra-d-adf-01.json
        linkedService
          ls_azdbw_existingcluster_01.json
          ls_azdbw_jobclusterv3_01.json
        pipeline
        publish_config.json
      techtf-sharedinfra-s-adf-01
    dbw
      dataproduct
        src
          setup
            install_dependencies.py
            pre_run_script.py
            __init__.py
          helloworld.py
          __init__.py
  templates
  .gitignore
```



TechTacoFriday

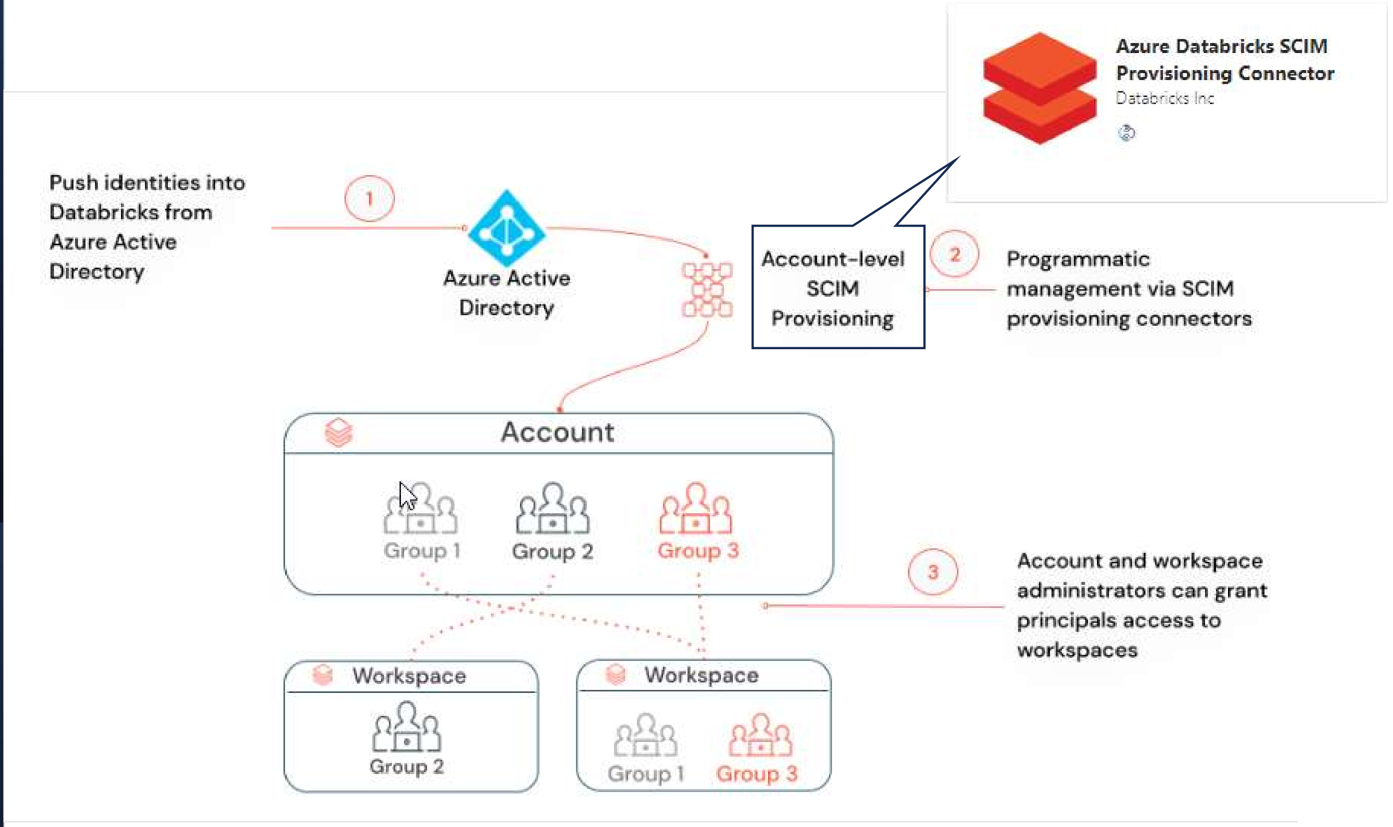
Technology spiced with a taco flavor mix

Manage Databricks Users with Azure's Entra ID.

System for Cross-domain Identity Management

SCIM
System for Cross-domain Identity Management
SCIM 2, the open API for managing identities is now complete and published under the IETF.

"[…] specification is designed to make managing user identities in cloud-based applications and services easier" Ref. https://scim.cloud/


Azure Databricks SCIM Provisioning Connector
Databricks Inc
Push identities into Databricks from Azure Active Directory
1
Azure Active Directory
Account-level SCIM Provisioning
2
Programmatic management via SCIM provisioning connectors
Account
Group 1
Group 2
Group 3
3
Account and workspace administrators can grant principals access to workspaces
Workspace
Group 2
Workspace
Group 1
Group 3

Configuring SCIM for the Account

Step 1. Check the Requirements

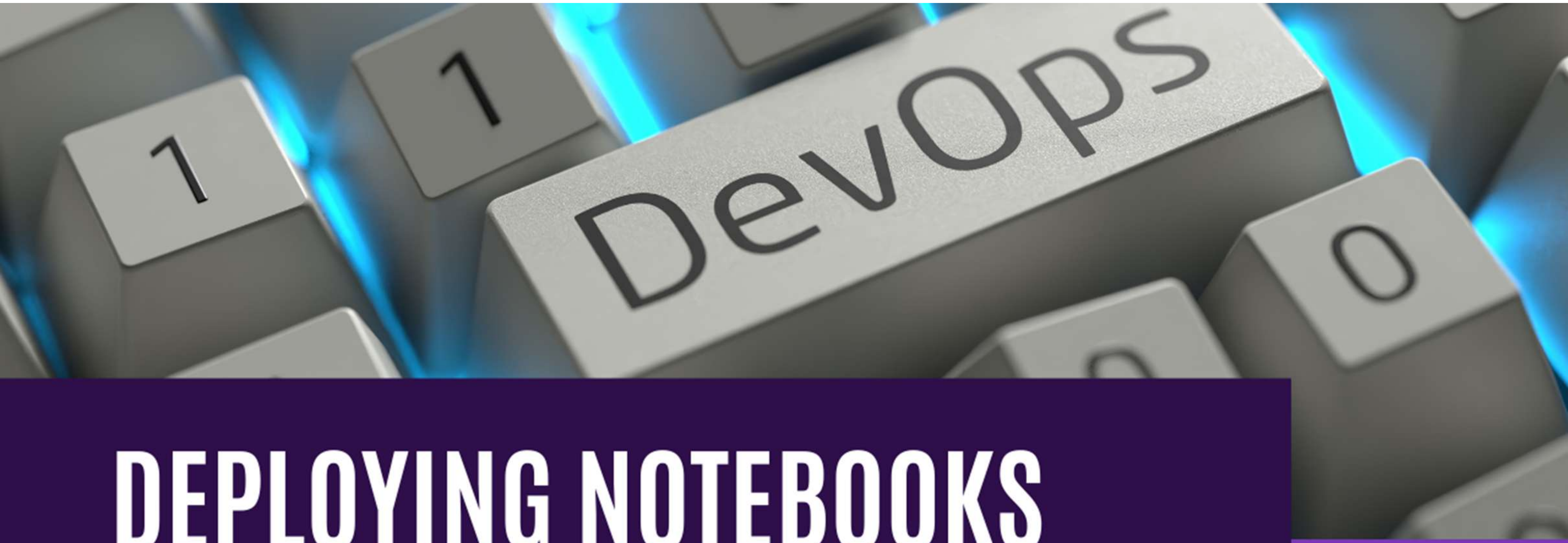
- ✓ Cloud Application Administrator role in Microsoft Entra ID
- ✓ You must be an Azure Databricks account admin
- ✓ Azure Databricks account must have the Premium plan

Step 2. Configure Azure Databricks Account

- ✓ Enable User Provisioning
- ✓ Copy Account SCIM URL & Token

Step 3. Create and configure SCIM application

- ✓ Create the SCIM Enterprise App
- ✓ Configure with Account’s SCIM URL & Token



DEPLOYING NOTEBOOKS WITH DEVOPS YAML PIPELINES.

TechTacoFriday

Technology spiced with a taco flavor mix

Quick poll … which one is your approach?

Method 1. Databricks Repos with Automated Sync

Repos (a feature in Databricks) are used to sync notebooks from a Git repository directly into a workspace.

Method 2. CI/CD Pipeline-Based Deployment

This approach involves integrating Databricks into a Continuous Integration/Continuous Deployment (CI/CD) pipeline, often using tools like Jenkins, GitHub Actions, Azure DevOps, or GitLab CI/CD. In this approach, the CI/CD pipeline orchestrates the promotion of notebooks (or other Databricks artifacts) from one environment to another.

Method 3. API-Based or CLI-Based Deployment

In this approach, deployments are managed via scripts that interact with Databricks through its REST API or CLI. This is similar to the CI/CD approach but can be more lightweight, focusing on custom scripts to promote changes between environments.

Method 4. Manual approach (aka Copy & Paste)

Copy & paste the content of my workbook from one environment to the other

ORCHESTRATE YOUR NOTEBOOKS WITH AZURE.

TechTacoFriday

Technology spiced with a taco flavor mix

Databricks Linked service & cluster selection

Job cluster

- ✓ Modular approach
- ✓ Job cluster can be reused in multiple executions
- ✓ Job cluster is terminated when the last chain execution is finished

Existing cluster

- ✓ Better for schedule tasks that happen regularly
- ✓ Cluster configuration can be stored and reused as well as libraries inhalations
- ✓ Time to live can be configured in the cluster

Existing instance pool

- ✓ Used when chained executions are needed
- ✓ Same pool can be used in different pipelines (be mindful on the concurrency limits)
- ✓ Lower start up times
- ✓ Job cluster can be configured to use the pool

Key Advantages of Azure Data Factory (ADF) vs. Databricks Orchestration

	ADF	Databricks
Broader Integration Ecosystem	**100+ cloud and on-premise data sources** (e.g., SQL Server, Oracle, SAP, Salesforce, Cosmos DB)	Primarily optimized for running Databricks notebooks and tasks within the Databricks ecosystem.
Data Movement & Hybrid Scenarios	Built-in support for **hybrid scenarios** (data migration from on-premises via Integration Runtime and gateways).	Less efficient and harder to manage when orchestrating large-scale data migration from on-premises sources.
Low-Code/No-Code Visual Pipeline Development	Visual, drag-and-drop, code-free authoring environment.	Requires notebook-based scripting (Python, Scala, SQL) or JSON-based task configuration
Separation of Concerns and Flexibility	Easier to **swap or augment compute technologies** in the future without major re-architecture	Replacing or augmenting compute platforms later can require significant rework.
When to Use Each: Quick Reference	✓ Complex workflows involving various Azure and third-party services ✓ Hybrid (cloud/on-premises) data movement scenarios	✓ Simple scheduling/execution of Databricks notebooks ✓ Pure Databricks ecosystems with minimal external interactions

Streamlining Databricks: CI/CD your Notebooks with DevOps Pipelines and orchestrate via Azure Data Factory (Series)

On this series I'm going to show you how to provision your Databricks infrastructure with BICEP and to connect your workspace to Azure's Entra ID to manage users & groups. Furthermore, I'll show you how to deploy your notebooks across Environments with yaml pipelines and orchestrate with ADF

Download the code here …

🔗 https://www.techtacofriday.com/orchestrate-your-notebooks-via-azure-data-factory/

Read the article series here

🔗 https://www.techtacofriday.com/streamlining-databricks-cicd-with-bicep-devops-and-adf/

Thank you for joining

A big shoutout to…

twoday

Executions and nice tricks

- ✓ Execute Azure databricks jobs in sequence or with conditions
  - Use the combination activities and Azure Databricks
    - Cluster reuse
    - Parametrization
    - Rerun
    - Delta live tables
    - Use pipelines as templates
    - Reuse code, triggers and infrastructure
- ✓ Pools and job clusters
  - Provide workload isolation
  - Reduces pricing
  - Auto termination
  - Faster job cluster creation