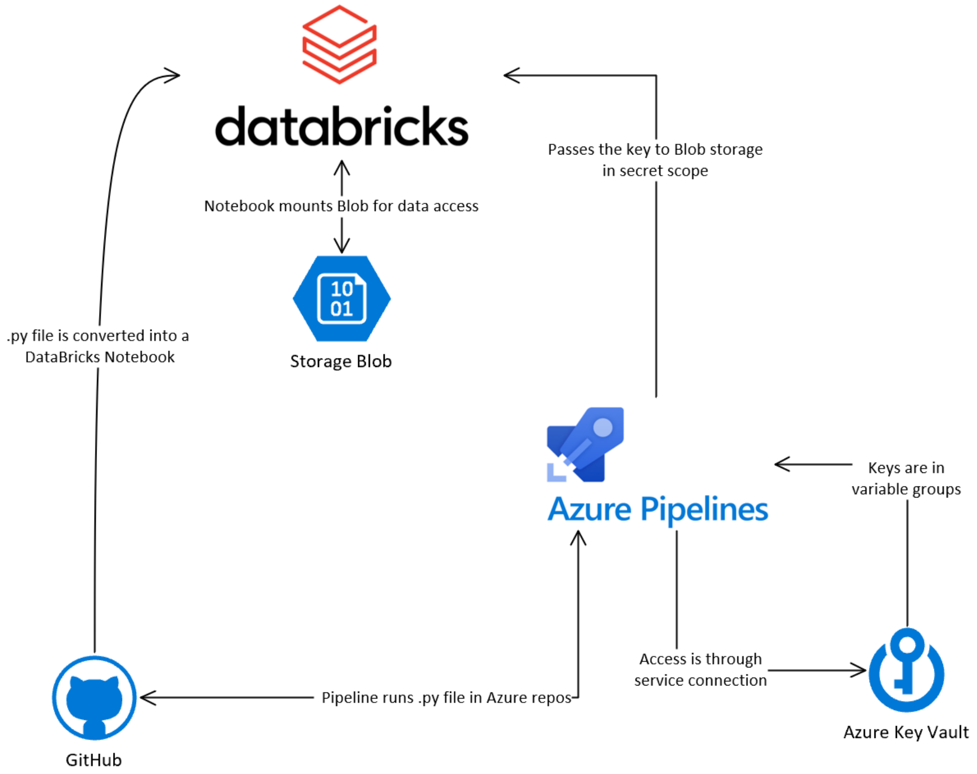
Table of Contents

An Accelerator to Provision Databricks with Azure DevOps

Objective

The purpose of this tutorial is a walkthrough to how one could securely set up Azure Databricks with an Azure Storage container mounted. Azure DevOps allows for an easy integration between all the different services required. Below is a visual representation of the overall architecture of the tutorial.


databricks
Passes the key to Blob storage in secret scope
Notebook mounts Blob for data access
10
01
Storage Blob
.py file is converted into a DataBricks Notebook
Azure Pipelines
Keys are in variable groups
Access is through service connection
Azure Key Vault
Pipeline runs .py file in Azure repos
GitHub

Prerequisites

For this project, the following prerequisites are assumed:

- Azure Active Directory Tenant (As in, can access Azure Portal)
- Azure Subscription
- GitHub account

Getting Started

For this section, the following resources should be created:

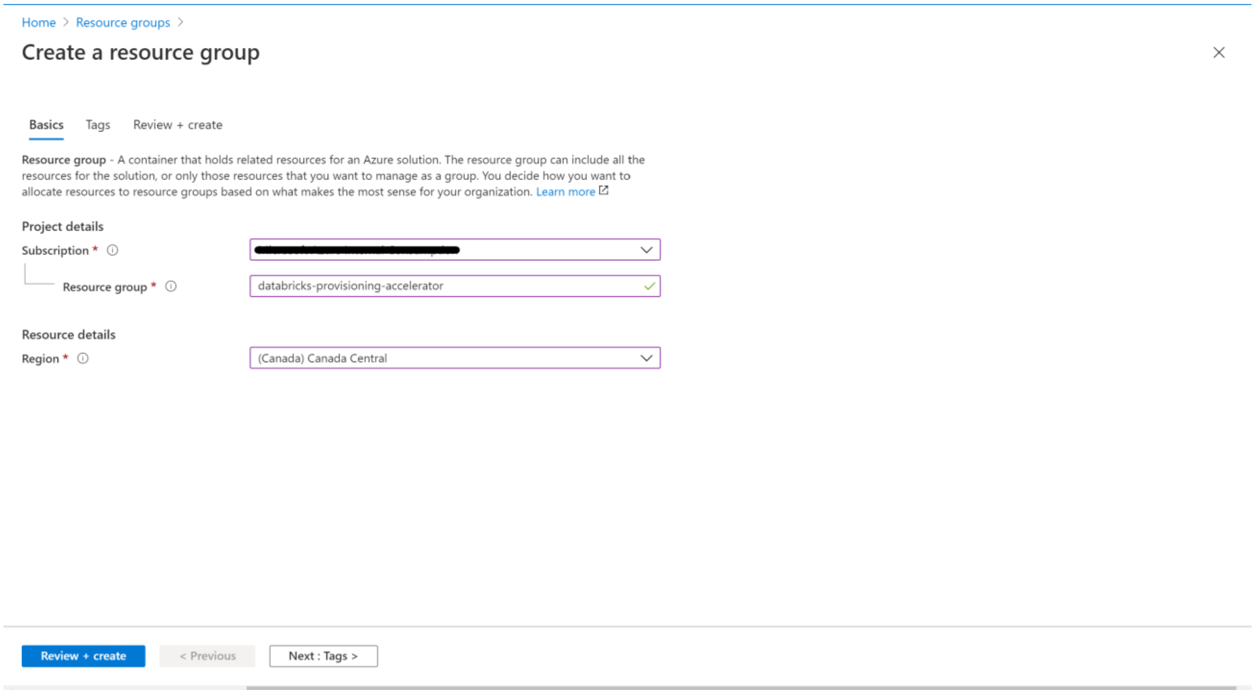
- Resource Group
- Storage Account
- Container in the storage account
- Key Vault with a secret
- Azure Databricks Service

Note: It is highly advisable to use the same names, but the user could choose different names. However, if the names are changed, make sure to change the respective names in right sections.

Resource	Name	Should be a different name?
Resource Group	databricks-provisioning-accelerator	NO
Storage Account	storageaccelerator	YES
Container in storage	storage-container	NO
Key Vault	acceleratorkeyvault	YES
Azure Databricks Service	accelerator-databricks	NO

Creating a Resource Group

1. Login into http://portal.azure.com/
2. In the search bar, search for **Resource Groups**
3. Click the *+ ADD* button
4. Fill in the required fields (as shown below)
5. Click *Review + Create*
6. Click *Create*



Create a resource group

Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. Learn more

Project details

Subscription * : [redacted]

Resource group * : databricks-provisioning-accelerator

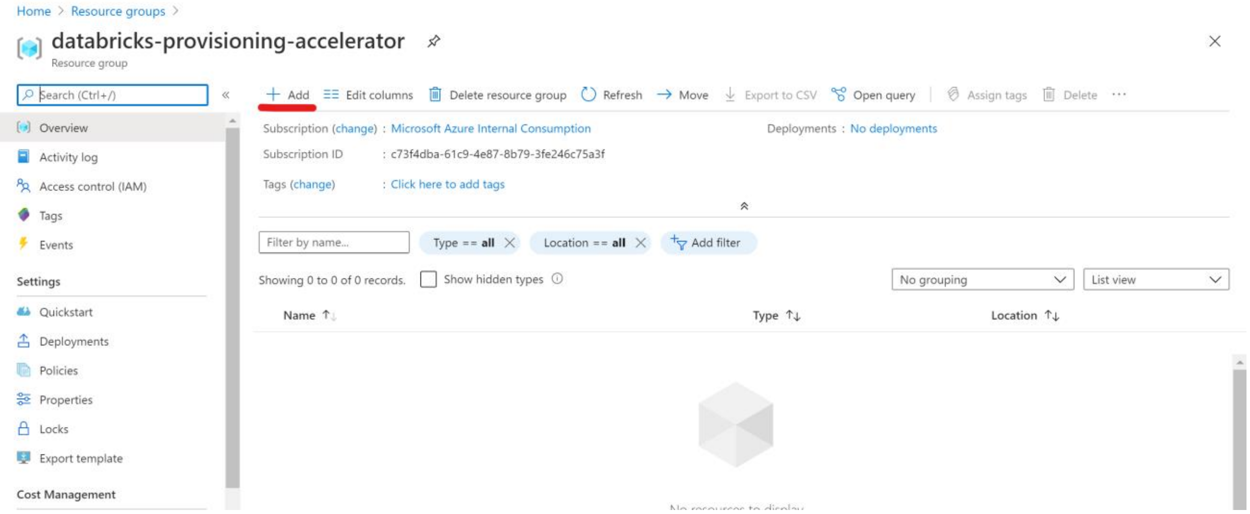
Resource details

Region * : (Canada) Canada Central

Creating a Storage Account

(Note: One could also provision Azure Data Lake Gen 2 (with hierarchical namespace enabled))

1. Open the Resource Group created previously
2. Click *+ ADD* (as shown below)



Subscription (change) : Microsoft Azure Internal Consumption

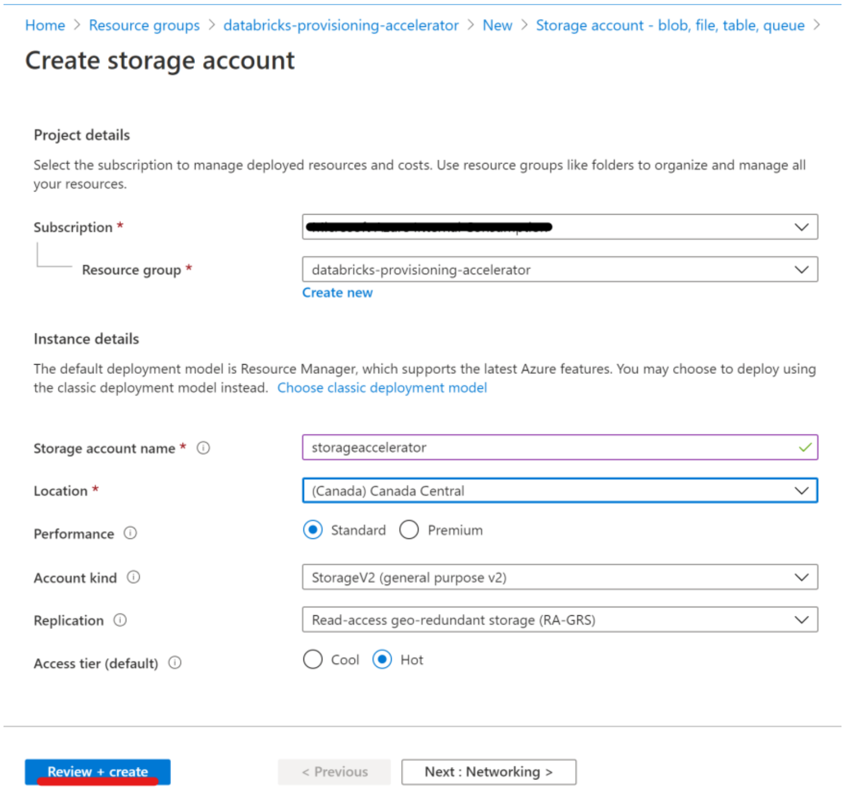
Subscription ID : c73f4dba-61c9-4e87-8b79-3fe246c75a3f

Tags (change) : Click here to add tags

Deployments : No deployments

Showing 0 to 0 of 0 records.

3. Search for **Storage Account** and click *Create*
4. Fill in the fields as required and click *Review + Create*
5. Click *Create*


Home > Resource groups > databricks-provisioning-accelerator > New > Storage account - blob, file, table, queue >
Create new
Choose classic deployment model
Review + create
< Previous
Next : Networking >

Create storage account

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * [redacted]

Resource group * databricks-provisioning-accelerator

Instance details

The default deployment model is Resource Manager, which supports the latest Azure features. You may choose to deploy using the classic deployment model instead.

Storage account name * storageaccelerator

Location * (Canada) Canada Central

Performance ● Standard ○ Premium

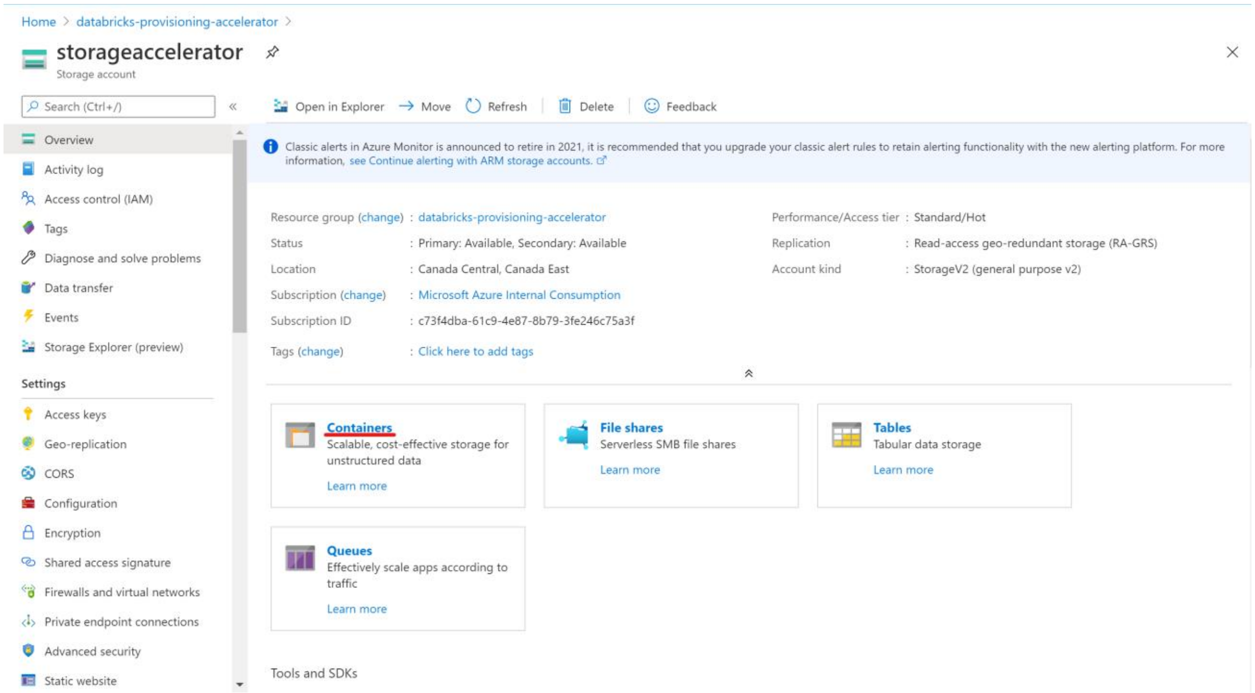
Account kind StorageV2 (general purpose v2)

Replication Read-access geo-redundant storage (RA-GRS)

Access tier (default) ○ Cool ● Hot

Creating a Container in a Storage Account

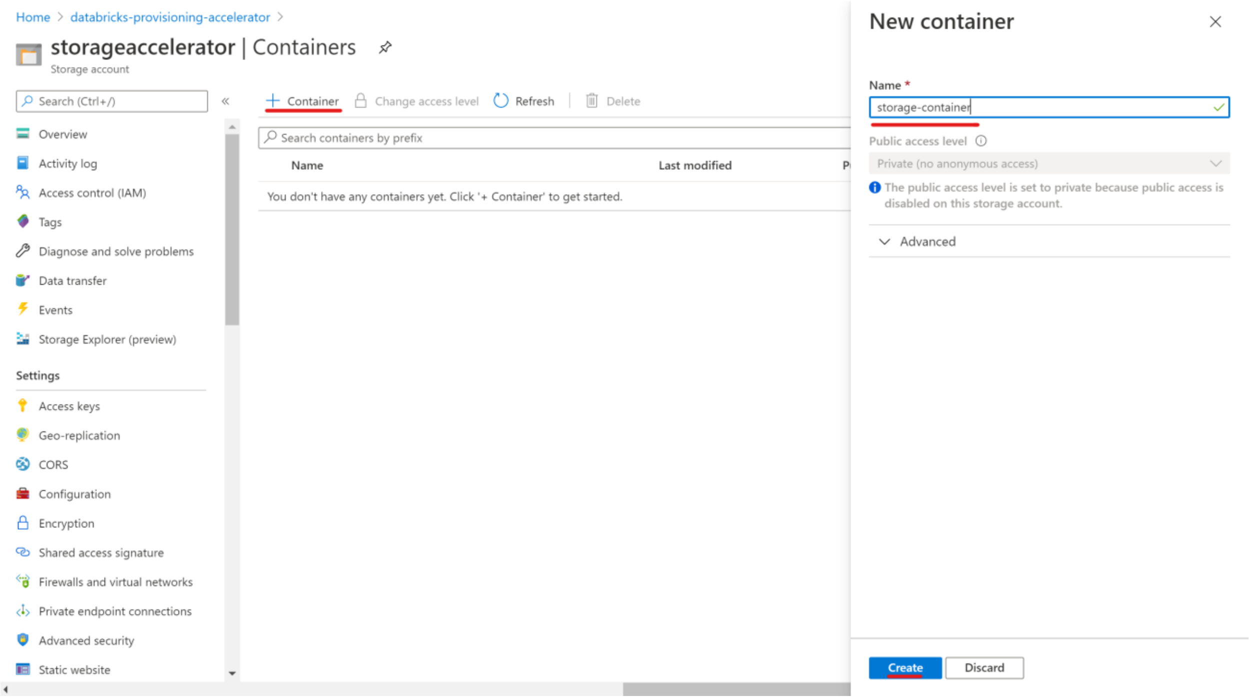
1. Open the Resource Group created earlier
2. Click on the storage account created in the previous step
3. Click on *Containers*



Classic alerts in Azure Monitor is announced to retire in 2021, it is recommended that you upgrade your classic alert rules to retain alerting functionality with the new alerting platform. For more information, see Continue alerting with ARM storage accounts.

Resource group (change)	: databricks-provisioning-accelerator	Performance/Access tier	: Standard/Hot
Status	: Primary: Available, Secondary: Available	Replication	: Read-access geo-redundant storage (RA-GRS)
Location	: Canada Central, Canada East	Account kind	: StorageV2 (general purpose v2)
Subscription (change)	: Microsoft Azure Internal Consumption		
Subscription ID	: c73f4dba-61c9-4e87-8b79-3fe246c75a3f		
Tags (change)	: Click here to add tags		

4. Click on *+ Container*
5. Fill the *Name* field and click *Create*



You don't have any containers yet. Click '+ Container' to get started.

Name *: storage-container

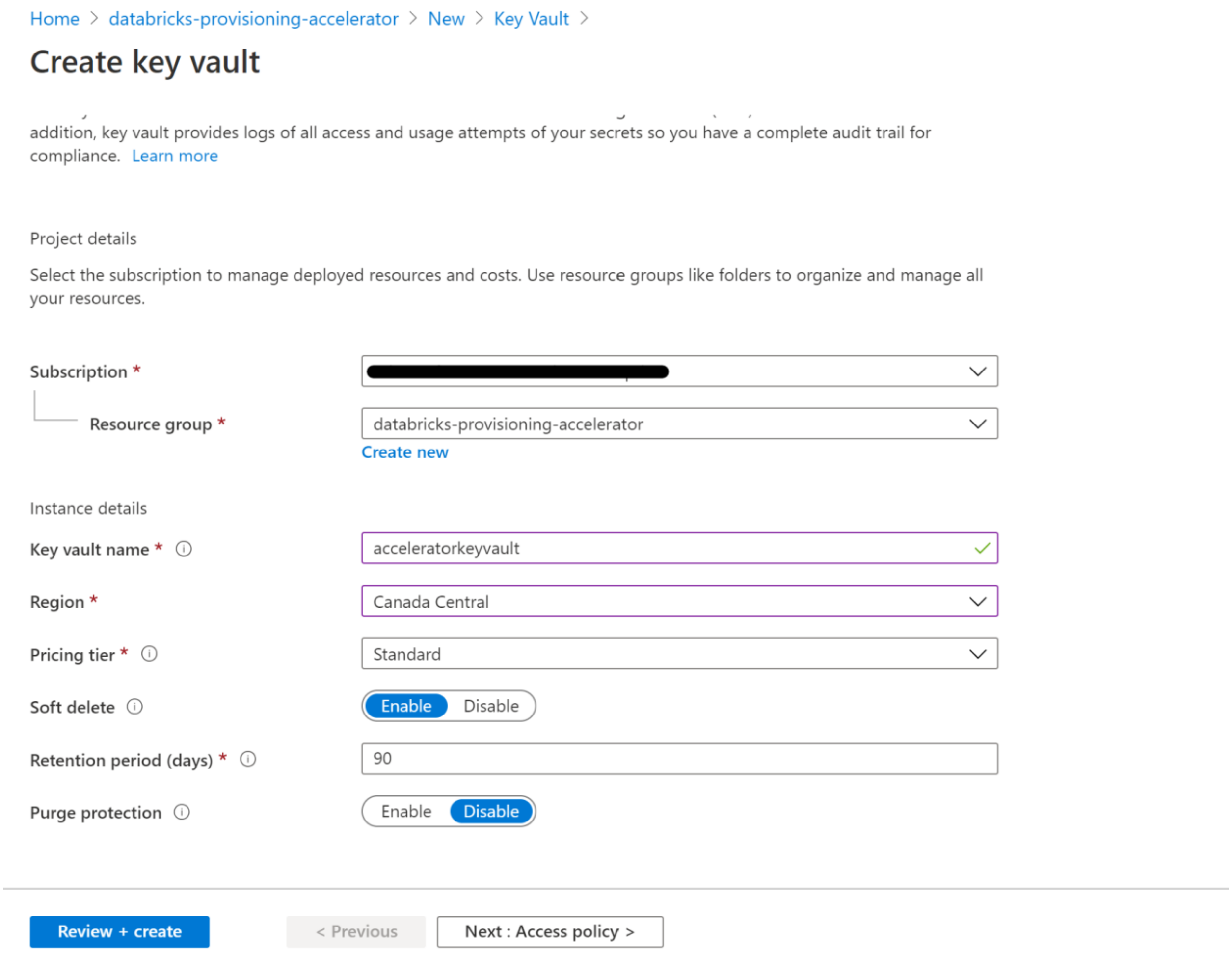
Public access level: Private (no anonymous access)

The public access level is set to private because public access is disabled on this storage account.

Creating a Key Vault account with a Secret

For more info click here

1. Open the Resource Group created earlier
2. Click *+ ADD*
3. Search for **Key Vault** and click *Create*
4. Fill in the fields as required and click *Review + Create*



Home > databricks-provisioning-accelerator > New > Key Vault >

Create key vault

addition, key vault provides logs of all access and usage attempts of your secrets so you have a complete audit trail for compliance. Learn more

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * [redacted]

Resource group * databricks-provisioning-accelerator

Create new

Instance details

Key vault name * ⓘ acceleratorkeyvault ✓

Region * Canada Central

Pricing tier * ⓘ Standard

Soft delete ⓘ [✓ Enable] Disable

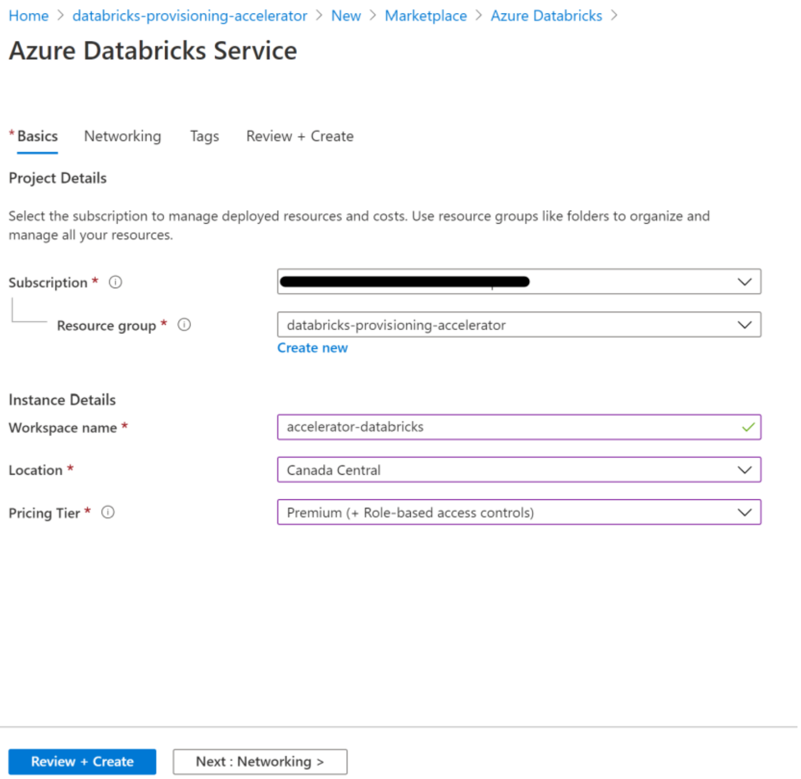
Retention period (days) * ⓘ 90

Purge protection ⓘ Enable [✓ Disable]

5. Click *Create*
6. Go to the Resource Group and open *acceleratorkeyvault*
7. On the left menu, under Settings, click *Secrets*
8. Click *+ Generate/Import*
9. Fill in the fields.
   a. **Note:** the key must be **accelerator-storage-key**, otherwise it will need to be changed in the pipeline's YAML file
   b. *Value* is retrieved from *storageaccelerator* > *Access keys* > copy *key1*
   c. For more info click here
10. Click *Create*

Creating an Azure Databricks Service

1. Open the Resource Group created earlier
2. Click *+ ADD*
3. Search for **Azure Databricks** and click *Create*
4. Fill in the fields. Note that the *Pricing Tier* should be *Premium*


Home > databricks-provisioning-accelerator > New > Marketplace > Azure Databricks >

Azure Databricks Service

* Basics
Networking
Tags
Review + Create

Project Details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * [redacted]

Resource group * databricks-provisioning-accelerator

Instance Details

Workspace name * accelerator-databricks

Location * Canada Central

Pricing Tier * Premium (+ Role-based access controls)

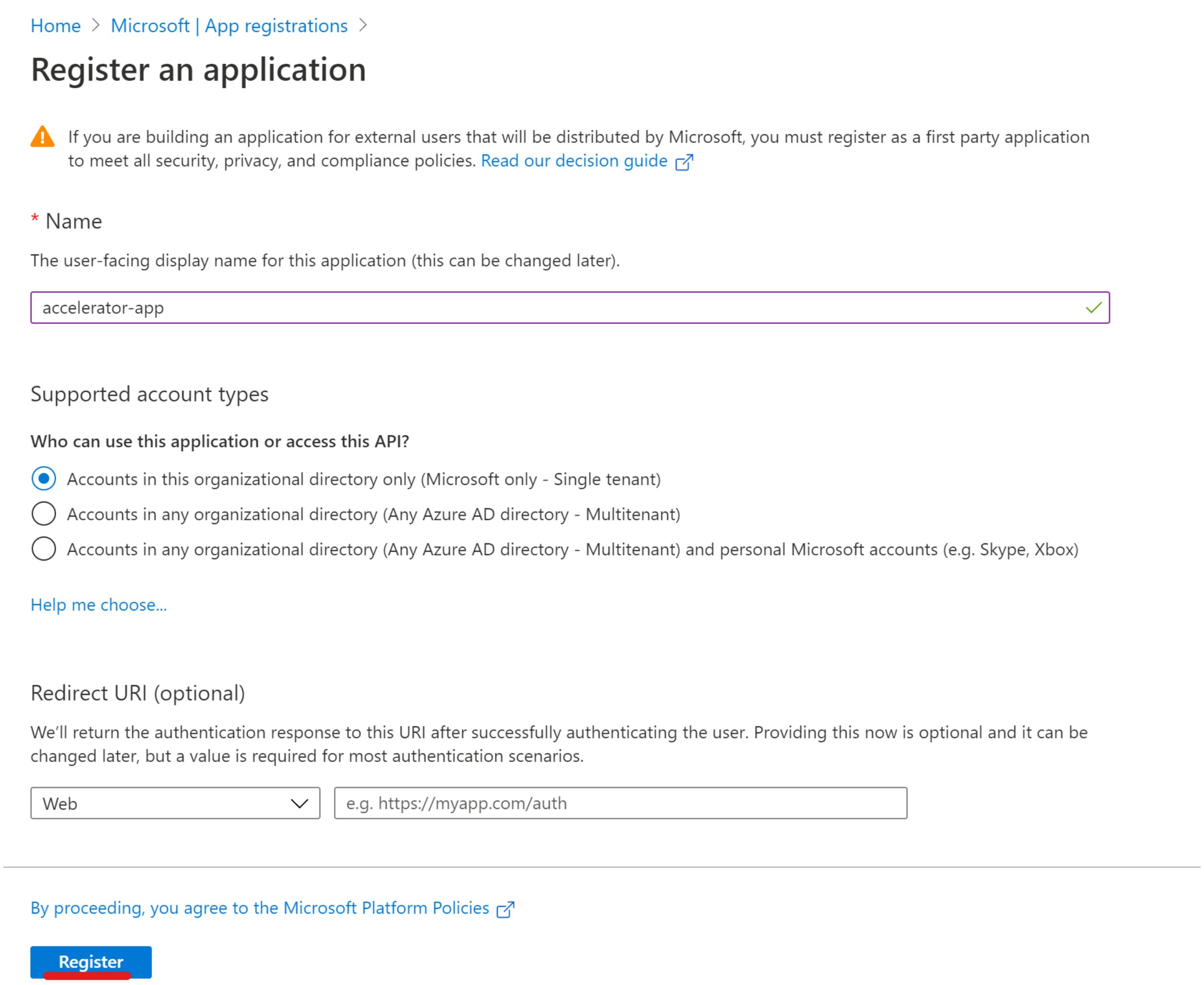
5. Click on *Review + Create*
6. Click *Create*

Steps to Set-up Databricks

1. Creating an App registration

For more information, click here.

1. Sign in to the Azure portal
2. Search for and select *Azure Active Directory*
3. Select *App Registrations*
4. Select *New Registration*
5. Fill the required fields as shown below and click *Register*



Home > Microsoft | App registrations >

Register an application

If you are building an application for external users that will be distributed by Microsoft, you must register as a first party application to meet all security, privacy, and compliance policies. Read our decision guide

* Name

The user-facing display name for this application (this can be changed later).

accelerator-app ✓

Supported account types

Who can use this application or access this API?

● Accounts in this organizational directory only (Microsoft only - Single tenant)

○ Accounts in any organizational directory (Any Azure AD directory - Multitenant)

○ Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)

Help me choose...

Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web | e.g. https://myapp.com/auth

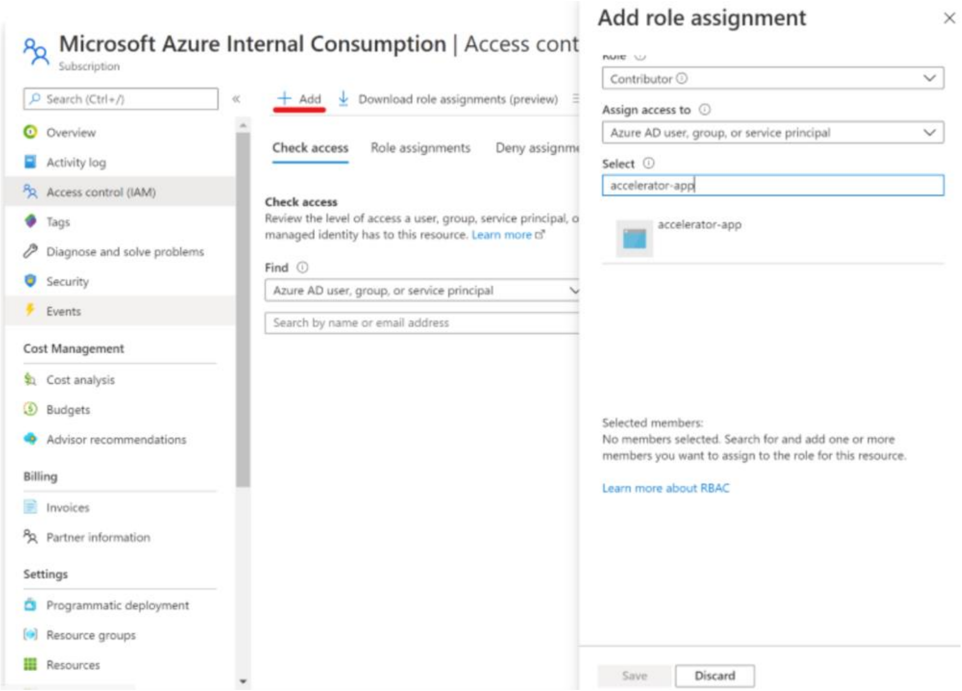
By proceeding, you agree to the Microsoft Platform Policies

Register

2. Assign a role to the application

For more information, click here.

1. In Azure portal, search for and select *Subscriptions*
2. Select the subscription you wish to use
3. Select *Access control (IAM)*
4. Select *Add Role Assignment*
5. Set the Role as *Contributor.* (Note: this is a high-privileged setting, one can give the service principal access to the resource group or even to the individual services.)
6. Select the application created previously and click *Save*



Check access

Review the level of access a user, group, service principal, o managed identity has to this resource. Learn more

Find: Azure AD user, group, or service principal

Role: Contributor

Assign access to: Azure AD user, group, or service principal

Select: accelerator-app

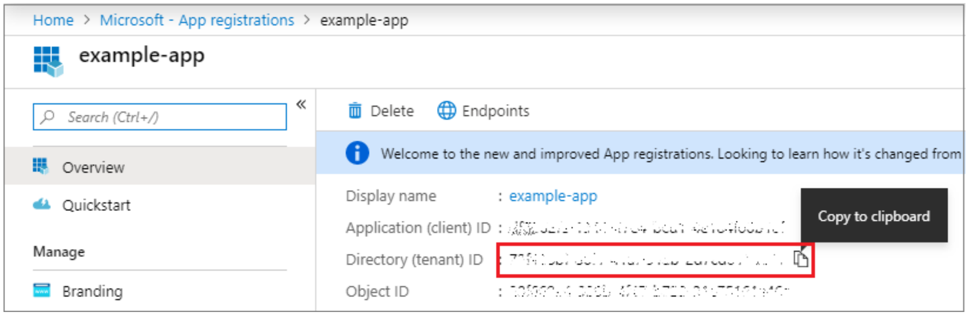
accelerator-app

Selected members:
No members selected. Search for and add one or more members you want to assign to the role for this resource.

3. Get the tenant and app ID values

We need to retrieve the tenant and app ID values for the next step. Therefore, store them somewhere safe, as they are needed later. For more information, click here.

1. In Azure portal, search and select *App Registrations*
2. Select the app created earlier
3. Copy the *Application (client) ID* and then the *Directory (tenant) ID*



Welcome to the new and improved App registrations. Looking to learn how it's changed from

Display name : example-app

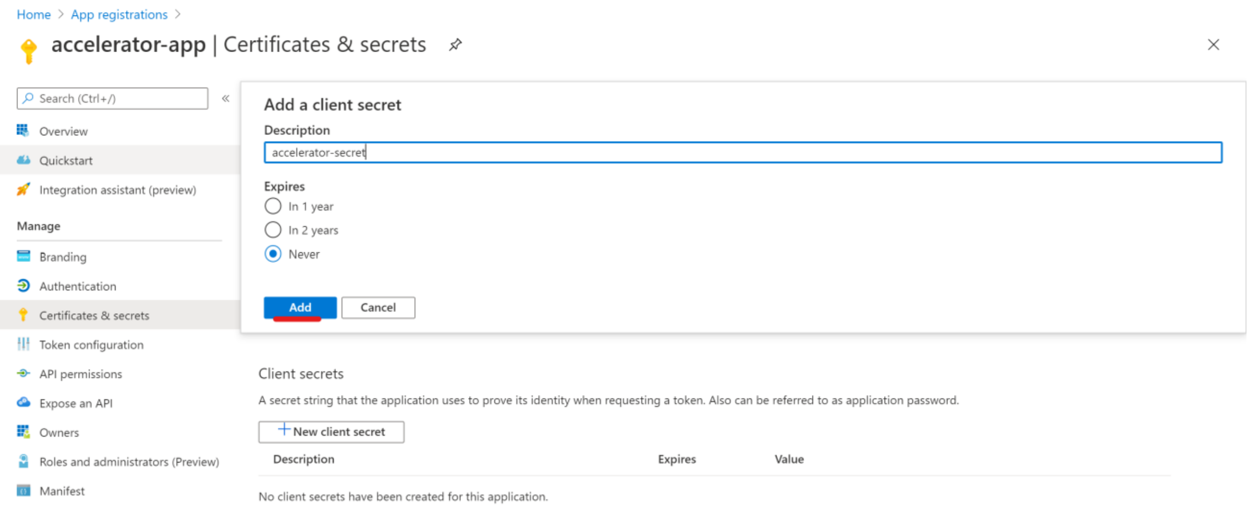
Application (client) ID : [redacted]

Directory (tenant) ID : [redacted]

Object ID : [redacted]

4. Creating an application secret

1. In Azure portal, search and select *App Registrations*
2. Select the app created earlier
3. Navigate to *Certificates & secrets*
4. Click *+ New client secret*
5. Fill in the required fields
6. Click *Add*
7. <span style="color:red">**Copy the value of the secret and store it securely (it can only be seen once)**</span>



Add a client secret

Description: accelerator-secret

Expires
○ In 1 year
○ In 2 years
● Never

Client secrets

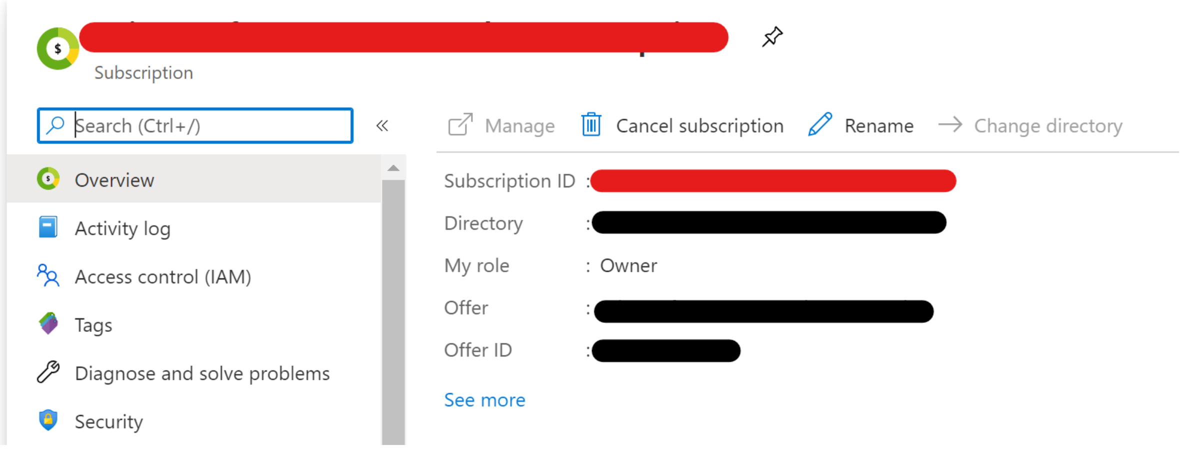
A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

Description	Expires	Value

No client secrets have been created for this application.

5. Retrieve Subscription Name and ID

1. In Azure portal, search and select *Subscriptions*
2. Select the subscription used throughout this project
3. Copy the subscription's name and subscription ID and store them safely



[redacted]
Subscription

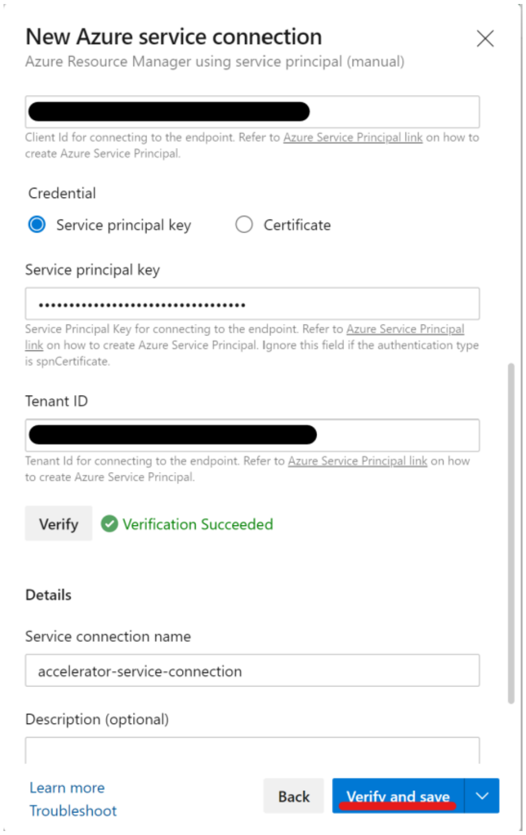
Subscription ID : [redacted]
Directory : [redacted]
My role : Owner
Offer : [redacted]
Offer ID : [redacted]

By now you should have the following hash values (necessary for the next step):

- Application (client) ID
- Directory (tenant) ID
- Value of the app's secret
- Subscription name and ID

6. Creating an Azure DevOps Project and a Service Connection

1. Login into https://dev.azure.com/
2. Click *+ Create Project*
3. Fill in the required fields and click *Create*
4. At the bottom of left menu bar, click on the gear logo for the Project Settings
5. Under *Pipelines*, click *Service Connections*
6. Click on *New service connection*
7. Click on *Azure Resource Manager*
8. Click on *Service principal (manual)*
9. Paste the Subscription ID and Name under *Subscription Id* and *Subscription Name* fields
10. Paste the Application (client) ID under the *Service Principal Id* field
11. Paste the value of the app's secret under *Service principal key*
12. Paste the Directory (tenant) ID under the *Tenant ID*
13. Click *Verify*, it's necessary to have a Verification Succeeded response
14. Fill in the *Service connection name* field
15. Click *Verify and save*



New Azure service connection

Azure Resource Manager using service principal (manual)

[redacted]

Client Id for connecting to the endpoint. Refer to Azure Service Principal link on how to create Azure Service Principal.

Credential

● Service principal key ○ Certificate

Service principal key

•••••••••••••••••••••••••••••••••

Service Principal Key for connecting to the endpoint. Refer to Azure Service Principal link on how to create Azure Service Principal. Ignore this field if the authentication type is spnCertificate.

Tenant ID

[redacted]

Tenant Id for connecting to the endpoint. Refer to Azure Service Principal link on how to create Azure Service Principal.

Verify ✓ Verification Succeeded

Details

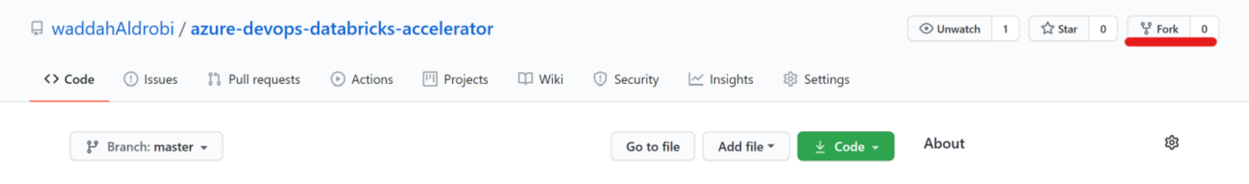
Service connection name

accelerator-service-connection

Description (optional)

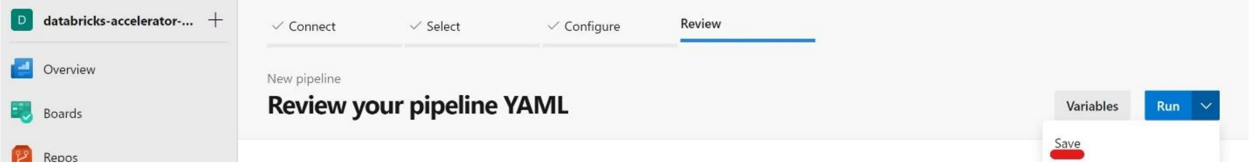
7. Forking the GitHub Repository

1. Navigate and sign in to https://github.com/
2. Navigate to https://github.com/waddahAldrobi/azure-devops-databricks-accelerator
3. Click on *Fork* to fork the repository on the master branch (as shown below)



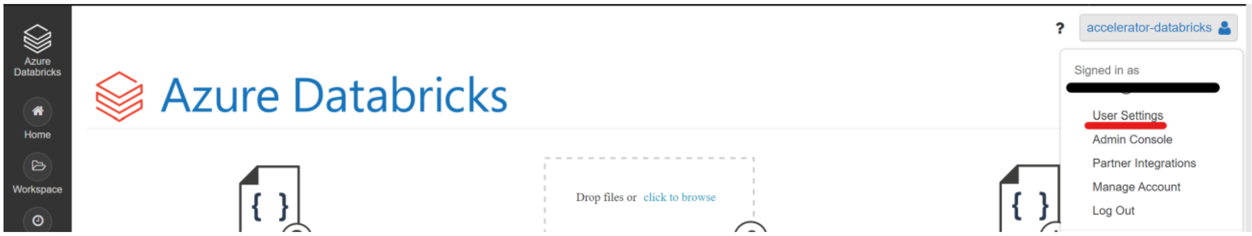
8. Creating an Azure DevOps Pipeline

1. Login into https://dev.azure.com/
2. Navigate to the project created earlier
3. In the project, navigate to *Pipelines*
4. Click *Create Pipeline*
5. Select *GitHub*
6. Select the GitHub repository from the previous step (follow any consequent instructions)
7. Click the arrow next to *Run* and click *Save*



9. Retrieving Databricks Personal Access Token and Region URL

1. Sign in to the [Azure portal](https://portal.azure.com)
2. Navigate to Resource Groups and select the resource group created earlier
3. Select the Databricks resource (*accelerator-databricks)*
4. Click on *Launch Workspace*
5. In the top right corner, select the name of the workspace, then select *User Settings*



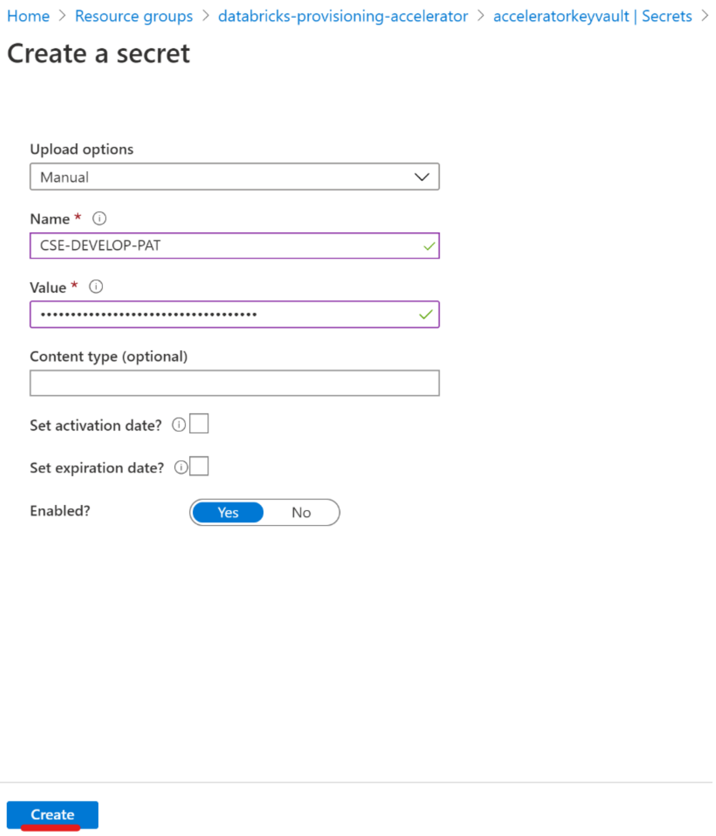
6. Under *Access Tokens,* click *Generate Token*
7. Fill in the required fields and click *Generate*
8. **Copy the token and store it securely.** (Should be in the form *dapi****) and click *Done*
9. Copy the part of URL in your browser that's in the form *https://***.azuredatabricks.net*

By now you should have the following hash values (necessary for the next step):

- The personal access token from Azure Databricks
- The workspace region URL

10. Adding the Personal Access Token to Key vault

1. Go to Azure Portal
2. Go to the Resource Group and open *acceleratorkeyvault* (or what you name the resource)
3. On the left menu, under Settings, click *Secrets*
4. Click + *Generate/Import*
5. Fill in the fields.
   a. **Note:** the key must be **CSE-DEVELOP-PAT**, otherwise it will need to be changed in the pipeline's YAML file
   b. Set the *Value* as the personal access token has value retrieved earlier
6. Click *Create*



Home > Resource groups > databricks-provisioning-accelerator > acceleratorkeyvault | Secrets >

Create a secret

Upload options: Manual

Name *: CSE-DEVELOP-PAT

Value *: ••••••••••••••••••••••••••••••••••••••

Content type (optional):

Set activation date? □

Set expiration date? □

Enabled? [✓ Yes] No

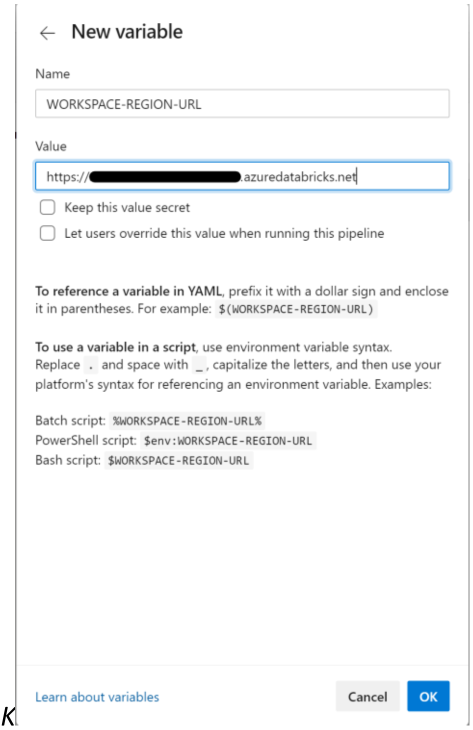
Create

11. Adding Variables to the Pipeline

1. Login into https://dev.azure.com/
2. Navigate to the project created earlier
3. Navigate to *Pipelines* and select the pipeline created earlier
4. In the top right corner, click *Edit*
5. In the top right corner, click *Variables*

Note: the following variables MUST have the same names, otherwise they will need to be changed in the pipeline's YAML file

6. Click *New Variable*
    a. Set *Name* as **WORKSPACE-REGION-URL**
    b. Set *Value* as the workspace region URL copied earlier



New variable

Name

WORKSPACE-REGION-URL

Value

https://[redacted].azuredatabricks.net

□ Keep this value secret

□ Let users override this value when running this pipeline

To reference a variable in YAML, prefix it with a dollar sign and enclose it in parentheses. For example: `$(WORKSPACE-REGION-URL)`

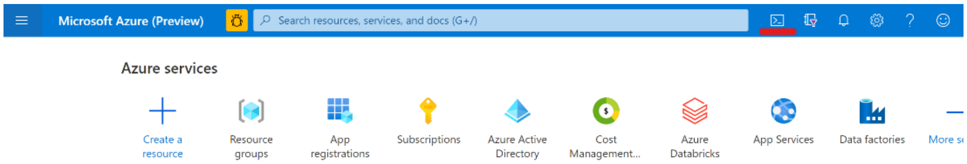
To use a variable in a script, use environment variable syntax. Replace `.` and space with `_`, capitalize the letters, and then use your platform's syntax for referencing an environment variable. Examples:

Batch script: `%WORKSPACE-REGION-URL%`
PowerShell script: `$env:WORKSPACE-REGION-URL`
Bash script: `$WORKSPACE-REGION-URL`

7. Click *OK*
8. Click on the *+* next to the search bar
    a. Set *Name* as **NOTEBOOK_NAME**
    b. Set *Value* as **mount.py**
9. Click OK
10. Click on the *+* next to the search bar
    a. Set *Name* as **SECERET-SCOPE-NAME**
    b. Set *Value* as **accelerator-secret-scope**
11. Click OK
12. Click on the *+* next to the search bar
    a. Set *Name* as **SCOPE-KEY-NAME**
    b. Set *Value* as **accelerator-secret**
13. Click on *OK, then* Click on *Save*

12. Adding a Variable Group containing Key Vault variables

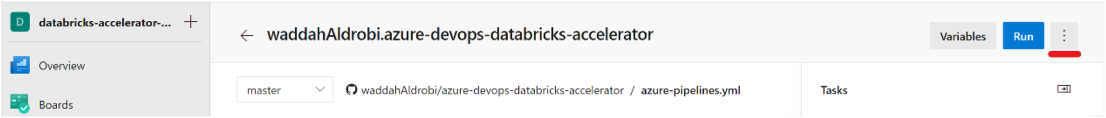
1. Using the menu bar on the left
2. Under *Pipelines,* navigate to *Library*
3. Click *+ Variable group*
4. T must be named **keyvault_variable_group**, else edit the YAML file
5. Click on the toggle *Link secrets from an Azure key vault as variables*
6. Select the service connection created earlier (accelerator-service-connection)
7. Select the Key vault name created earlier (acceleratorkeyvault)
8. When an error is prompted, copy the PowerShell script provided
9. Navigate to [Azure portal]()
10. Click on the cloud shell button as shown below



11. Make sure that a PowerShell environment is selected. Paste the script copied earlier and run it.
12. Follow the steps to authenticate.
13. To confirm, navigate to Resource Groups > <Project's resource group> > <Your Key Vault instance> > Access Policies. You should find your app's registration name under Applications.
14. Return to Azure DevOps, and click on the refresh button next to *Key vault name*
15. Under *Variables* click *+ Add* and select both *accelerator-storage-key* and *CSE-DEVELOP-PAT*
16. Click on *Save* (on the top bar)

13. Linking the Variable Group to the Pipeline

1. In Azure DevOps, click on *Pipelines*
2. Click on the pipeline created earlier
3. Click on *Edit* (Authorize if necessary)
4. Click on options (…)


databricks-accelerator-...
Overview
Boards
waddahAldrobi.azure-devops-databricks-accelerator
Variables
Run
master
waddahAldrobi/azure-devops-databricks-accelerator / azure-pipelines.yml
Tasks

5. Click on *Triggers*
6. Click on *Variables*
7. Click on *Variable Groups*
8. Click on *Link variable group*
9. Select the variable group created earlier
10. Click *Link*

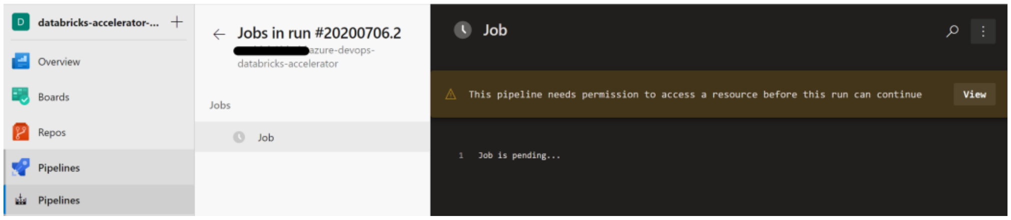
14. Running the Pipeline

(Note: If using Azure Datalake Gen 2, the content of mount.py should be changed. For more info, refer to https://docs.microsoft.com/en-us/azure/databricks/data/data-sources/azure/azure-datalake-gen2#--mount-an-azure-data-lake-storage-gen2-account-using-a-service-principal-and-oauth-20)

1. In Azure DevOps, click on *Pipelines*
2. Click on the pipeline created earlier
3. Click on *Run pipeline*
4. Click on *Run*

For the first time running the pipeline, permission to the service connection might be required:

5. Click on *Job*
6. Click on *View*
7. Click on *Permit*
8. Click on *Permit* to the popup



Jobs in run #20200706.2
[redacted]azure-devops-databricks-accelerator

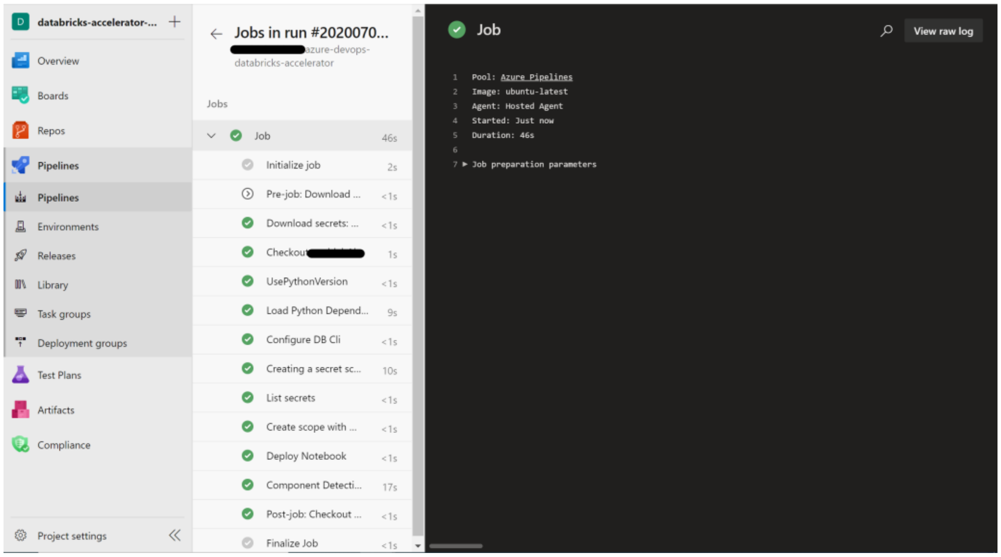
Job

Job

This pipeline needs permission to access a resource before this run can continue

```
Job is pending...
```

If everything has run successfully, the following is the result of pipeline's run:



Jobs in run #2020070...
[redacted]azure-devops-databricks-accelerator

Job	46s
Initialize job	2s
Pre-job: Download ...	<1s
Download secrets: ...	<1s
Checkout[redacted]	1s
UsePythonVersion	<1s
Load Python Depend...	9s
Configure DB Cli	<1s
Creating a secret sc...	10s
List secrets	<1s
Create scope with ...	<1s
Deploy Notebook	<1s
Component Detecti...	17s
Post-job: Checkout ...	<1s
Finalize Job	<1s

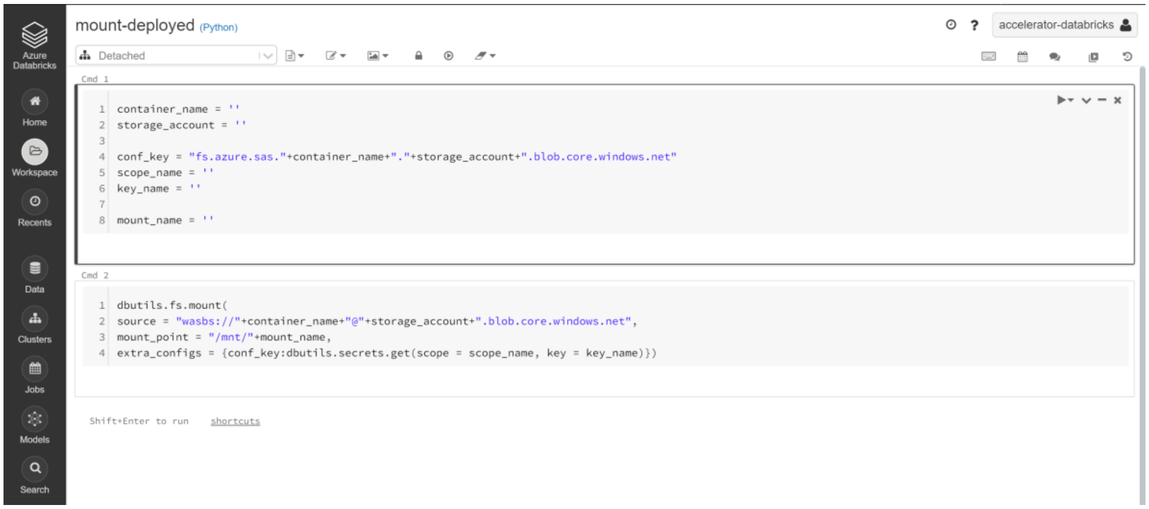
Job

```
Pool: Azure Pipelines
Image: ubuntu-latest
Agent: Hosted Agent
Started: Just now
Duration: 46s

▶ Job preparation parameters
```

15. Mounting the Databricks Notebook to Azure Storage

1. Sign in to the [Azure portal]
2. Navigate to Resource Groups and select the resource group created earlier
3. Select the Databricks resource (*accelerator-databricks)*
4. Click on *Launch Workspace*
5. Click *Home* from the left menu bar

6. Under *Workspace,* click on *Shared*
7. Click on *mount-deployed,* you should see the following:



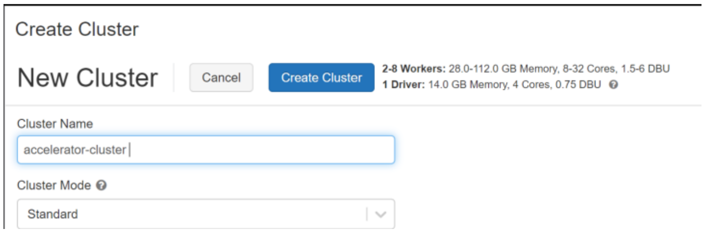
Cmd 1

```
container_name = ''
storage_account = ''

conf_key = "fs.azure.sas."+container_name+"."+storage_account+".blob.core.windows.net"
scope_name = ''
key_name = ''

mount_name = ''
```

Cmd 2

```
dbutils.fs.mount(
source = "wasbs://"+container_name+"@"+storage_account+".blob.core.windows.net",
mount_point = "/mnt/"+mount_name,
extra_configs = {conf_key:dbutils.secrets.get(scope = scope_name, key = key_name)})
```

8. Click on *Detached* and click on *Create a Cluster* (or choose a cluster if available)
9. Fill in the require fields and click *Create Cluster*

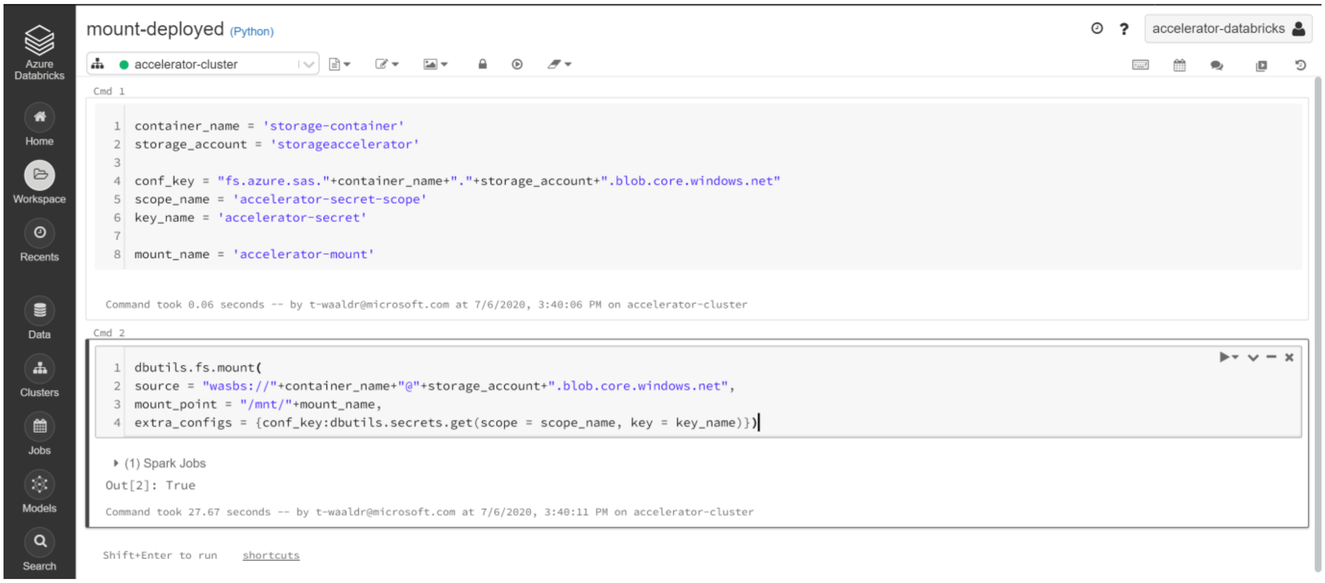


2-8 Workers: 28.0-112.0 GB Memory, 8-32 Cores, 1.5-6 DBU
1 Driver: 14.0 GB Memory, 4 Cores, 0.75 DBU

Cluster Name: accelerator-cluster

Cluster Mode: Standard

10. Wait for the cluster to run (take some time, about 5 minutes)
11. Return to the *mount-deployed* notebook
12. Click on *Detached* and attach the cluster created earlier
13. Fill in the variables with empty strings and run both cells (the output should be True)



Cmd 1

```
container_name = 'storage-container'
storage_account = 'storageaccelerator'

conf_key = "fs.azure.sas."+container_name+"."+storage_account+".blob.core.windows.net"
scope_name = 'accelerator-secret-scope'
key_name = 'accelerator-secret'

mount_name = 'accelerator-mount'
```

```
Command took 0.06 seconds -- by t-waaldr@microsoft.com at 7/6/2020, 3:40:06 PM on accelerator-cluster
```

Cmd 2

```
dbutils.fs.mount(
source = "wasbs://"+container_name+"@"+storage_account+".blob.core.windows.net",
mount_point = "/mnt/"+mount_name,
extra_configs = {conf_key:dbutils.secrets.get(scope = scope_name, key = key_name)})
```

```
▸ (1) Spark Jobs
Out[2]: True
Command took 27.67 seconds -- by t-waaldr@microsoft.com at 7/6/2020, 3:40:11 PM on accelerator-cluster
```