

Azure Data Factory + Azure DevOps

Objectives

- Implement DevOps Workflow ... git branches, pull requests, etc.

Technologies

This lab post assumes you have a Microsoft Azure subscription and pre-requisite knowledge about the following technologies:

- Azure Data Factory (<https://docs.microsoft.com/en-us/azure/data-factory/introduction>)
- Azure DevOps (<https://azure.microsoft.com/en-us/solutions/devops/>)
- GitHub (<https://github.com/>)

Step-by-Step Instructions

Instantiate Resources

First, we'll quickly run through creation of the basic resources we'll need to complete this exercise. You can use existing Azure resources in your subscription. If you don't have any of these resources already, please use below steps to provision resources.

Resource Group

<https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/manage-resource-groups-portal>

On the "Create a resource group" blade, populate the following:

- Subscription ... self-explanatory
- Resource Group ... choose a name that is meaningful for you (and aligned with your naming standards)
- Region ... select a region appropriate for your situation; take into consideration that some regions {e.g. West US and East US} see higher demand than others

Click the "Review + create" button and after validation, click the "Create" button. Allow time for processing.

Azure Data Factory

<https://docs.microsoft.com/en-us/azure/data-factory/quickstart-create-data-factory-portal>

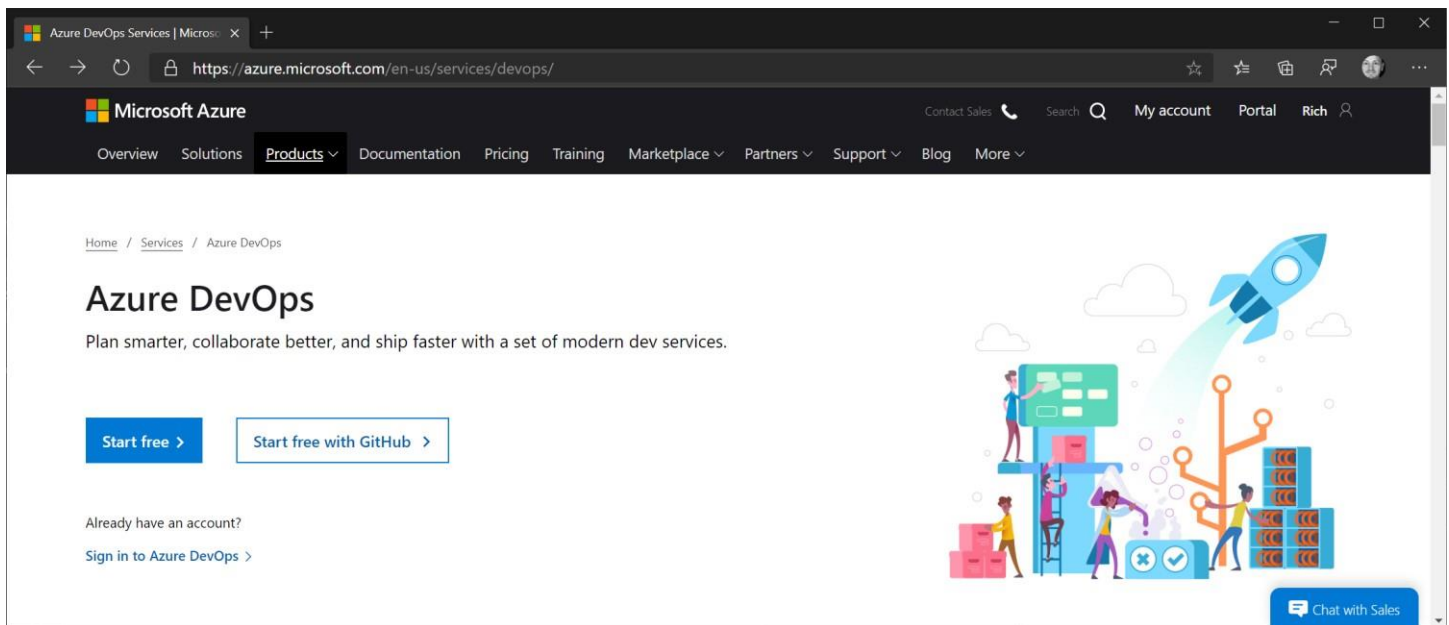
On the “New data factory” blade, populate the following:

- Name ... choose a name that is meaningful for you (and aligned with your naming standards)
- Version ... confirm default selection, “V2”
- Subscription ... self-explanatory
- Resource Group ... select the resource group created in the prior step
- Location ... select a region appropriate for your situation; take into consideration that some regions {e.g. West US and EastUS} see higher demand than others
- Enable Git ... we will, for now, un-check this box

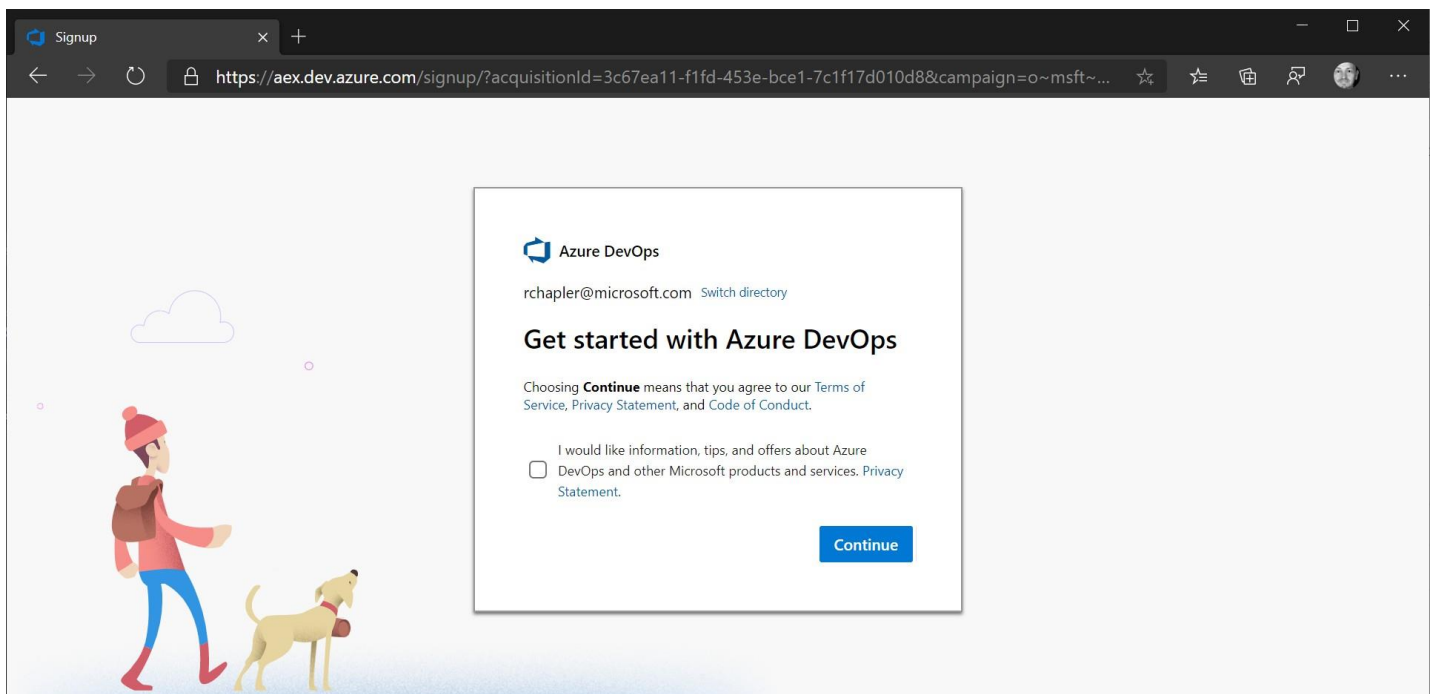
Click the “Create” button. Allow time for processing.

Azure DevOps

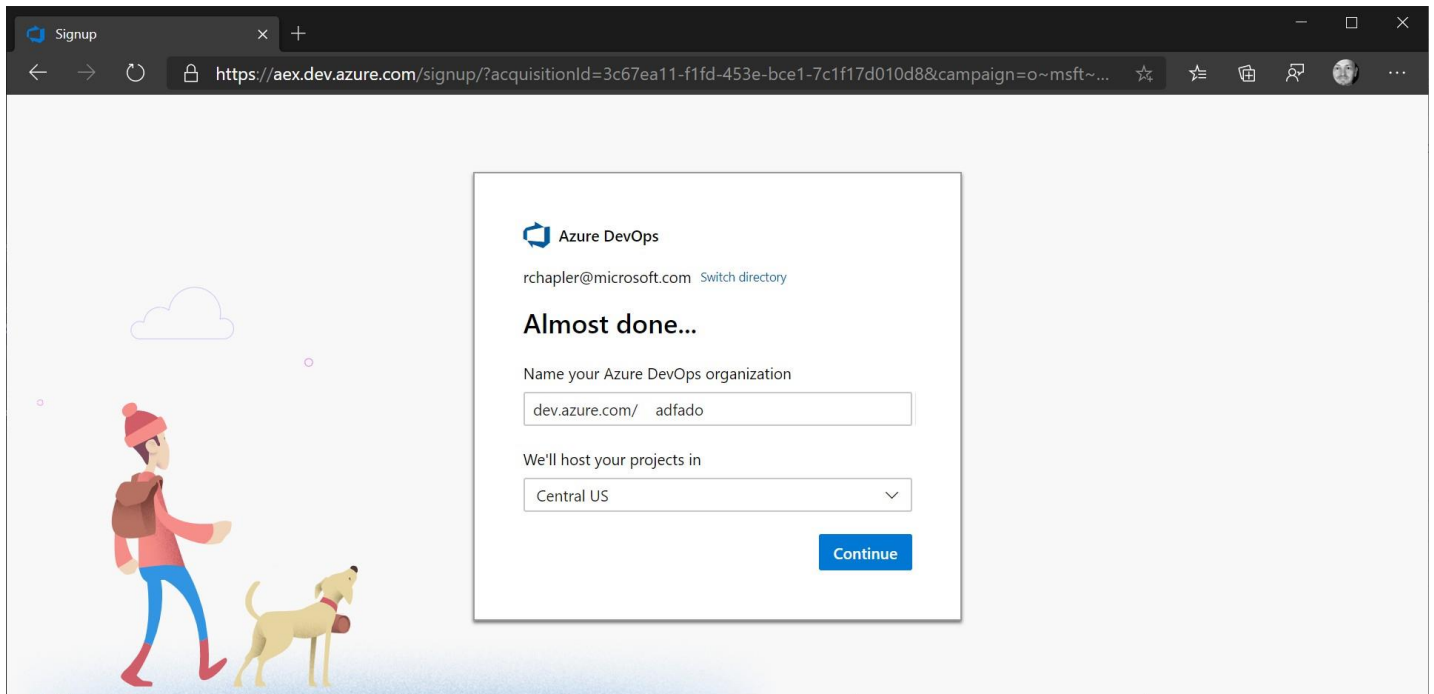
<https://azure.microsoft.com/en-us/services/devops/>



Click the “Start free with GitHub” button. On the resulting Azure DevOps page, click the “New organization” link on the left-hand navigation.



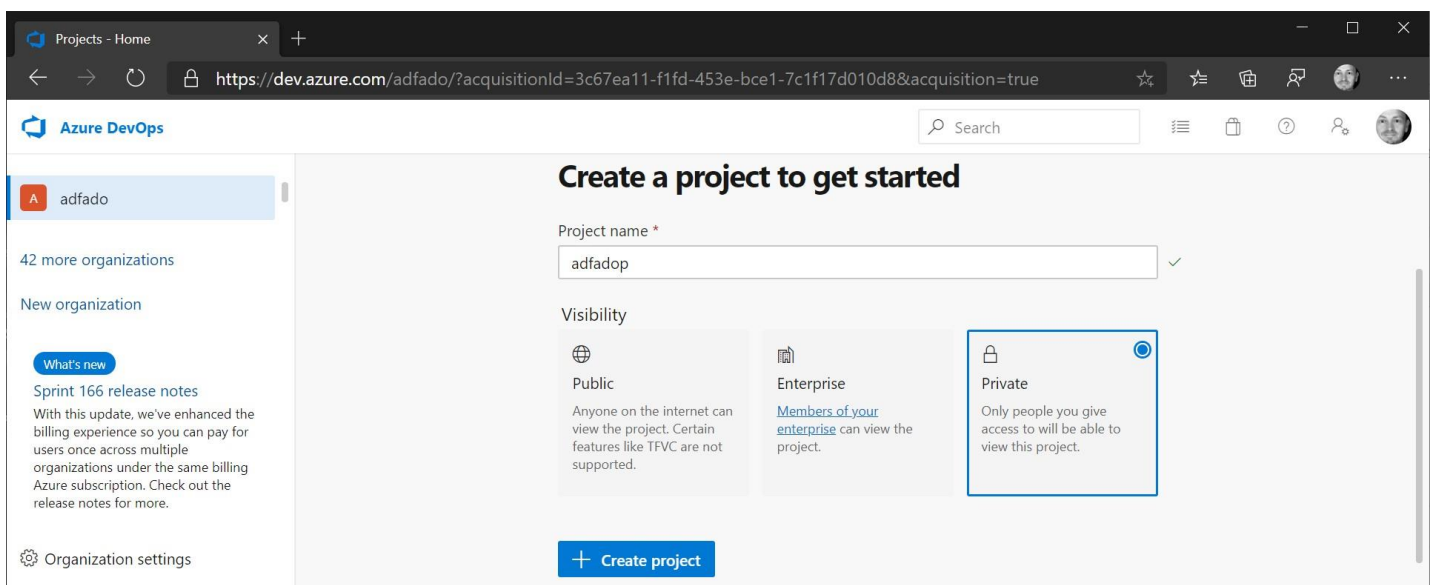
Confirm the directory and then click the “Continue” button.



On the “Almost done...” page, populate the following:

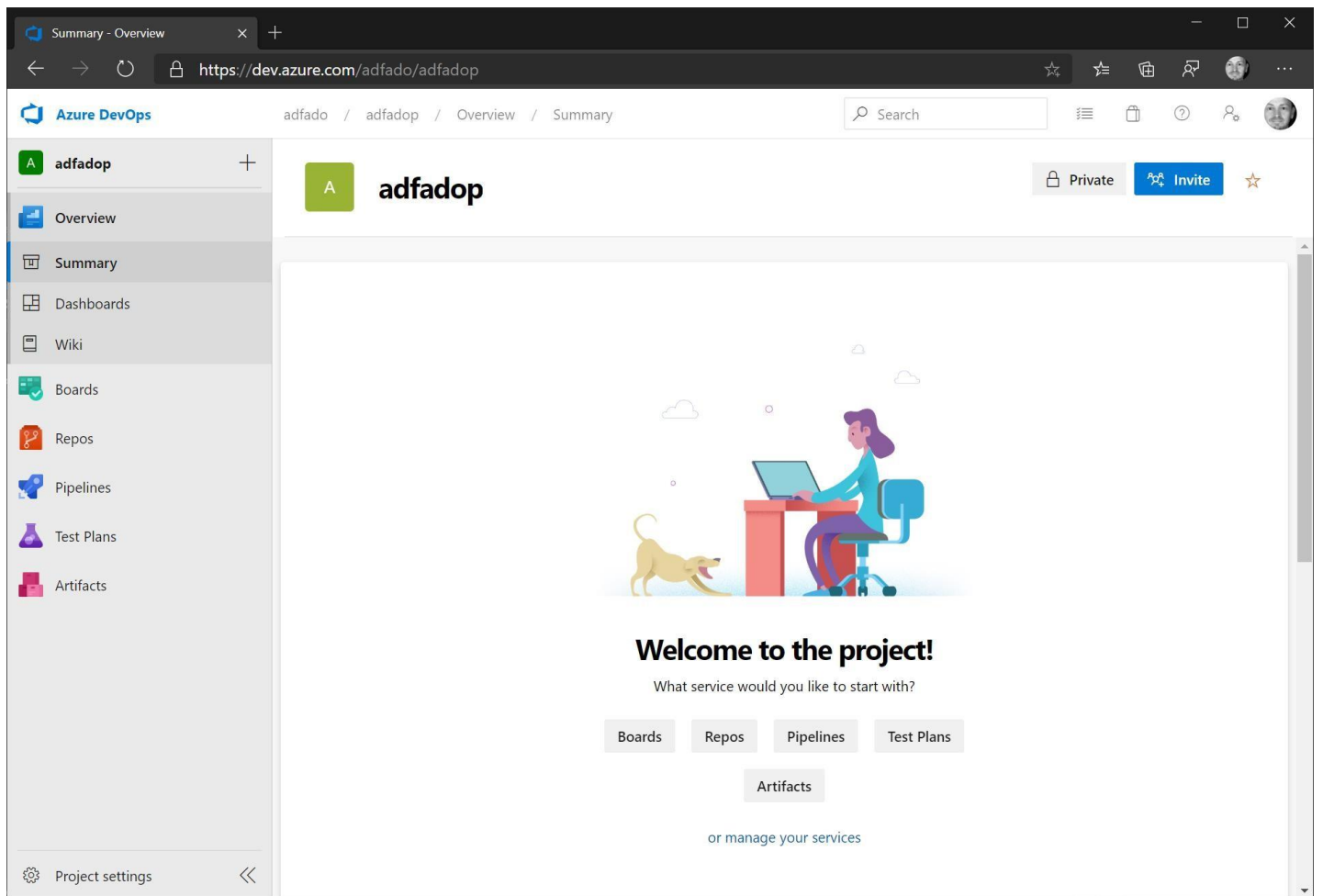
- Name your Azure DevOps organization ... choose a name that is meaningful for you (and aligned with your naming standards)
- We'll host your projects in ... select a region appropriate for your situation; take into consideration that some regions {e.g. West US and East US} see higher demand than others

Click the “Continue” button. Allow time for processing.



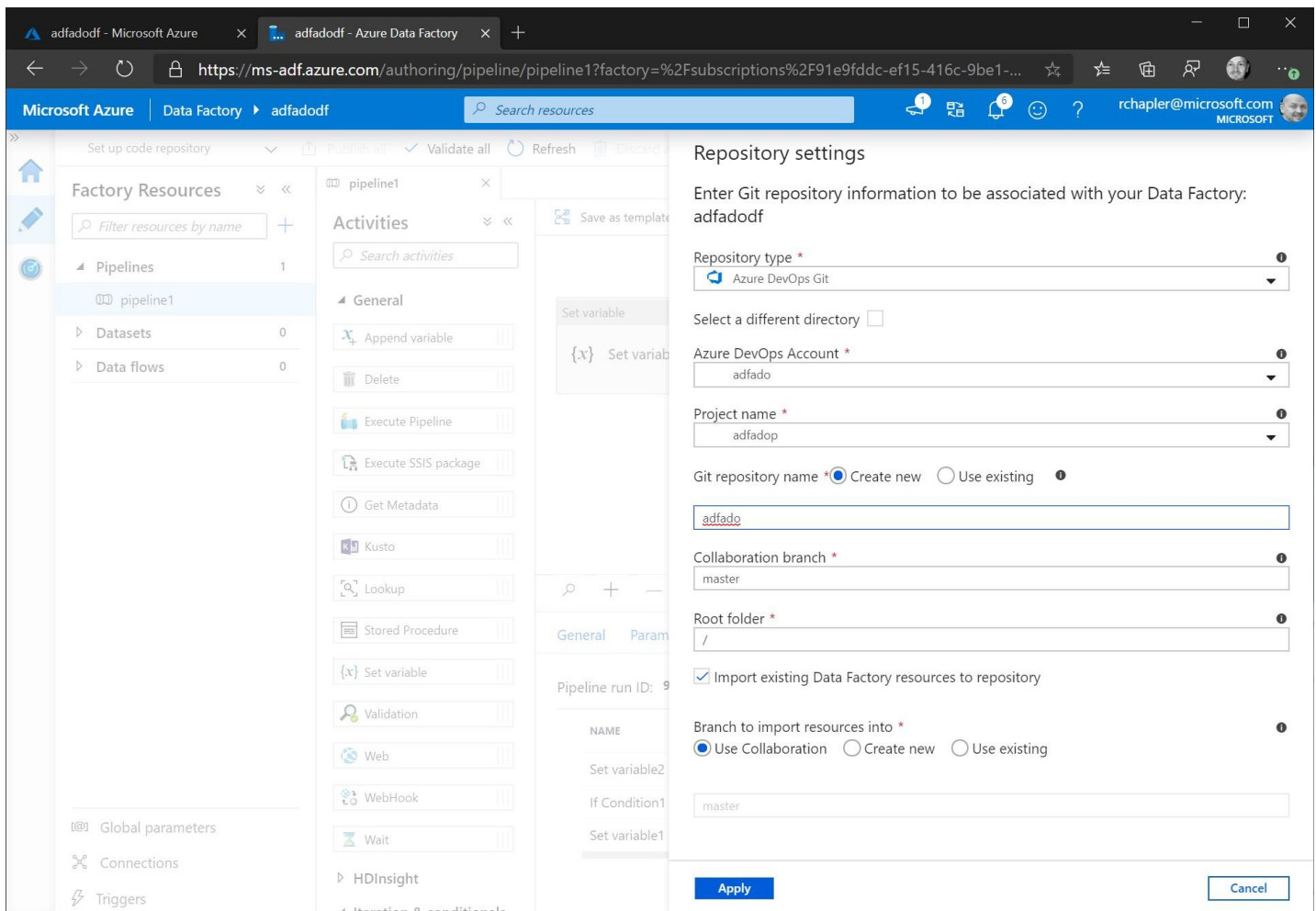
Click the “+ Create project” button. Allow time for processing.

When processing is complete, you will be navigated to a screen like the one snipped below.



Setup Code Repository

In the upper-right of the Data Factory window, click to dropdown “Data Factory” and select “Set up code repository”

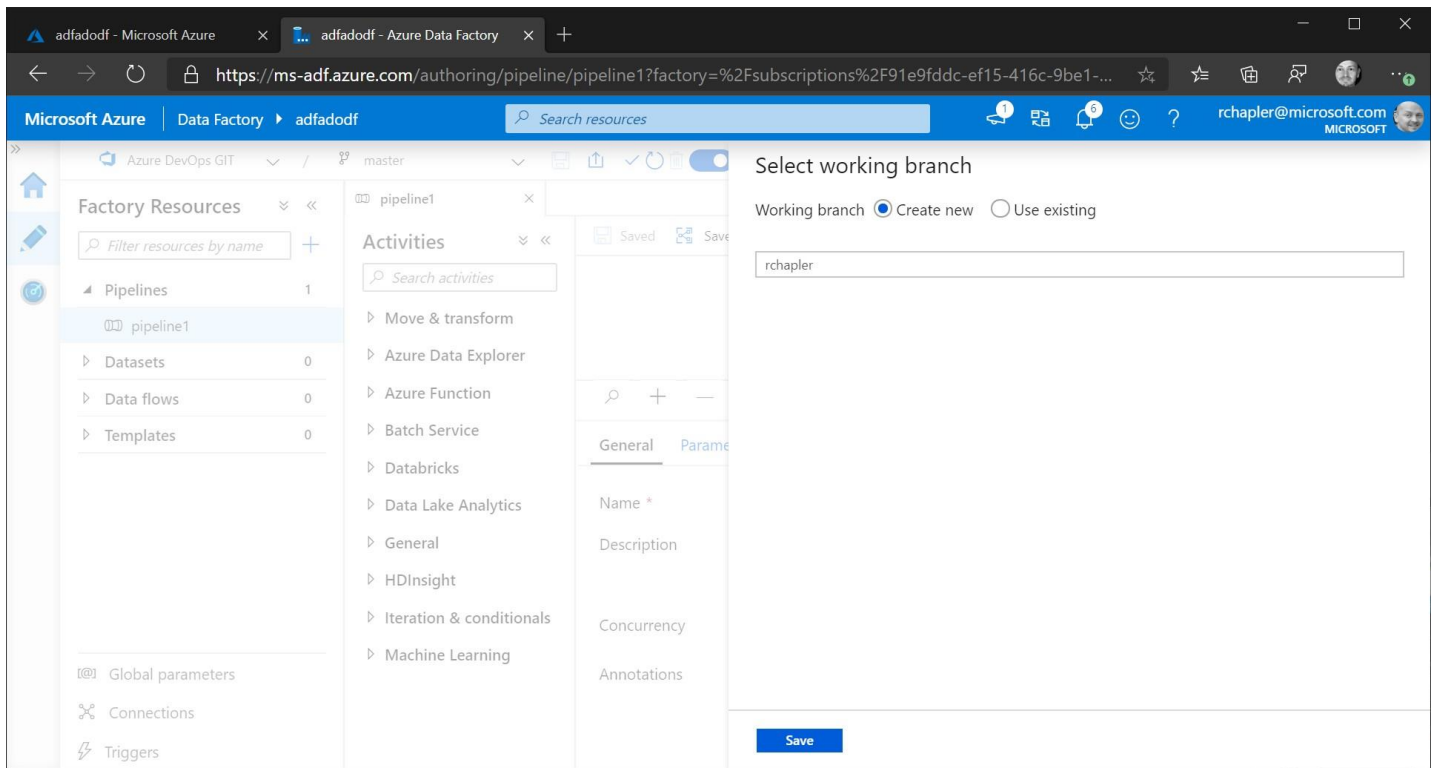


On the “Repository” popout, populate the following:

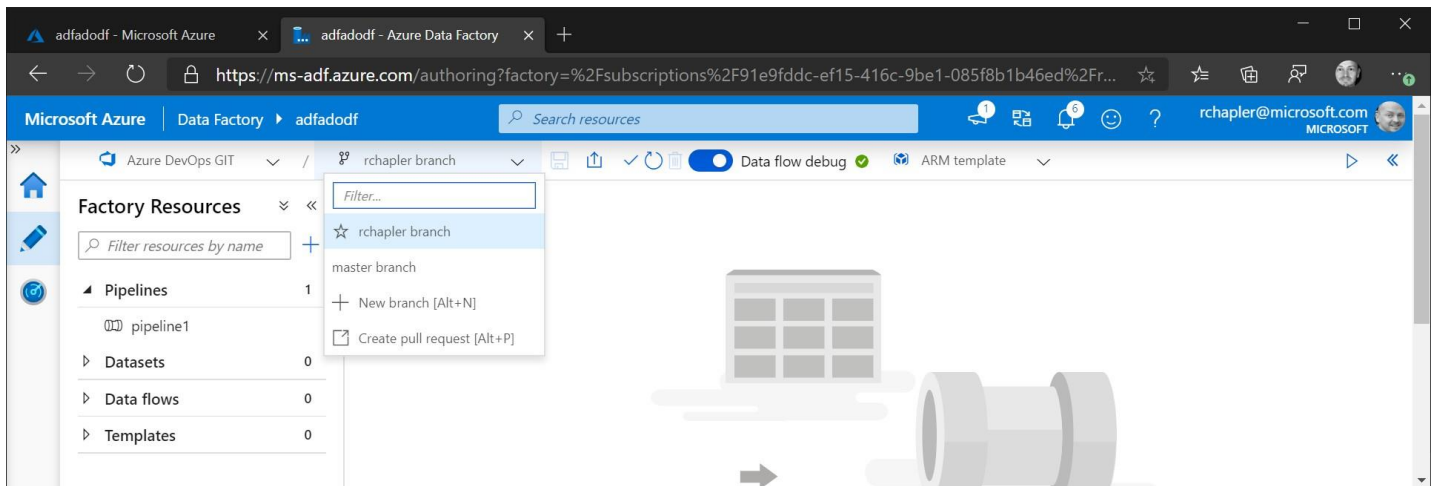
- Repository Type ... select Azure DevOps Git from the dropdown
- Azure DevOps Account ... select the DevOps instance created in the prior step
- Project name ... select the DevOps project created in the prior step
- Git repository name ... confirm default radio button selection, “Create new”, and then enter a name that is meaningful for you (and aligned with your naming standards)
- Azure DevOps Account ... select the DevOps instance created in the prior step
- Collaboration branch ... confirm default selection, “master”
- Root folder ... confirm default selection, “/”
- Import existing Data Factory resources to repository ... confirm default selection, checked
- Branch to import resources into ... confirm default selections, “Use Collaboration” radio button and “master” in

the textboxClick the “Apply” button. Allow time for processing.

Select working branch



In the resulting popout, select the “Create new” radio button and give your new branch a meaningful name. The branch name that you choose should depend on your branching strategy and development standards. In this case, I’m simply using my alias to let peer developers know that these are branched changes on which I’m working. Click the “Save” button. Allow time for processing.



When processing is complete, you’ll note that we are now working in the new branch. Everything developed to this point is already part of the master branch.

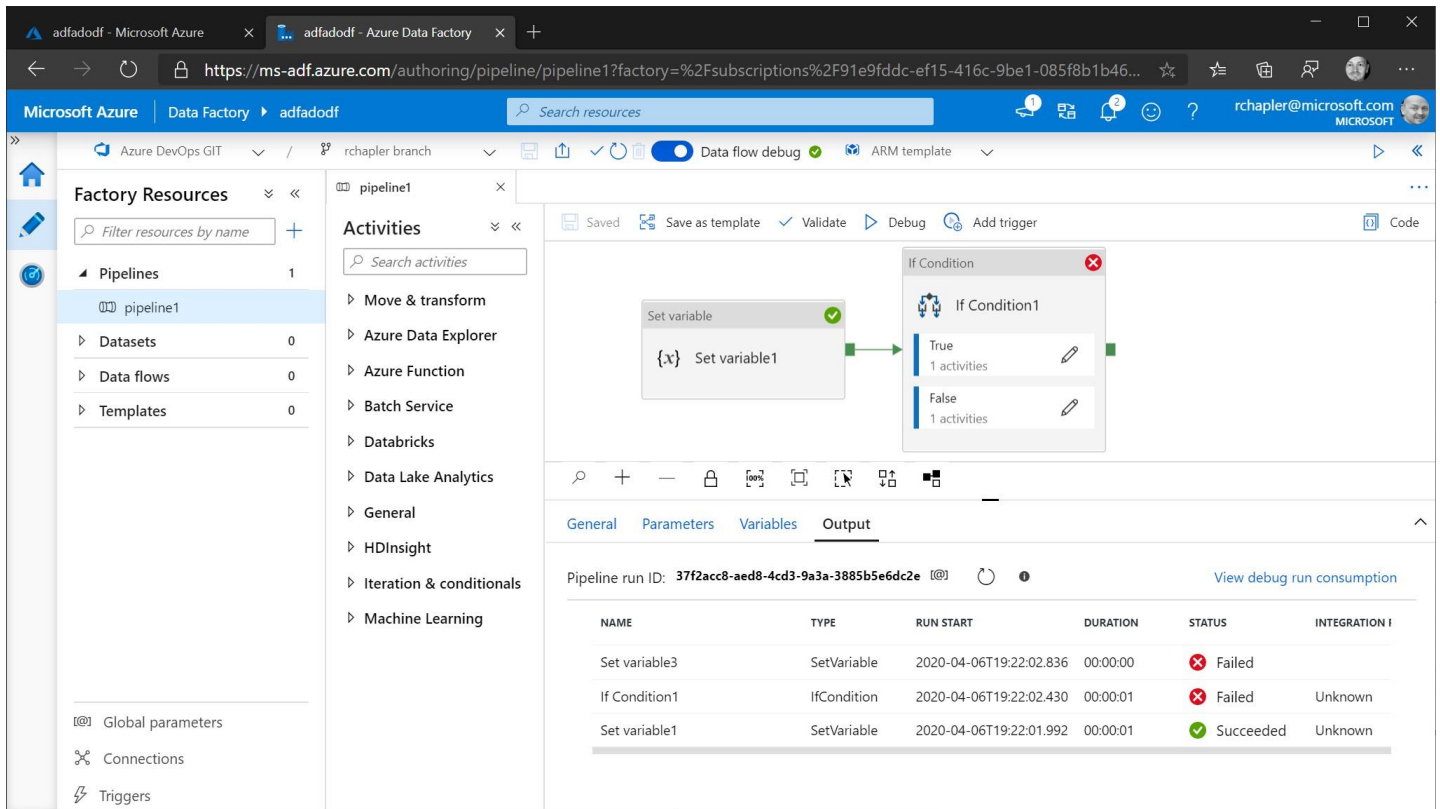
Modify Sample Application

We're going to make a minor change to the previously created pipeline in Lab 01, so we have something to process through DevOps.

The screenshot displays the Microsoft Azure Data Factory portal interface. The left sidebar shows the 'Factory Resources' tree with 'Pipelines' expanded, listing 'pipeline1'. The main canvas shows the 'pipeline1' diagram with a 'Set variable' activity (blue box) and an 'If Condition' activity (grey box). The 'Set variable' activity is configured with the variable name 'Set variable1'. The 'If Condition' activity is configured with the condition 'If Condition1'. Below the canvas, the 'Variables' tab is selected, showing the 'Name' as 'Flag' and the 'Value' as 'true'. The 'Value' field is highlighted, indicating it is the target for modification.

Navigate to “pipeline1” > “Set variable” > “Variables” tab and replace the “true” value with “false”.

Click the “Debug” button.



The screenshot shows the Azure Data Factory pipeline editor interface. The left sidebar displays 'Factory Resources' with a list of pipelines, datasets, data flows, and templates. The main canvas shows a pipeline named 'pipeline1' with two activities: 'Set variable' and 'If Condition'. The 'Set variable' activity is marked with a green checkmark, indicating it succeeded. The 'If Condition' activity is marked with a red 'X', indicating it failed. The 'If Condition' activity is configured with a condition that evaluates to 'False', leading to a failure. The 'Output' tab at the bottom shows the pipeline run details, including the pipeline run ID, a table of activities, and their status.

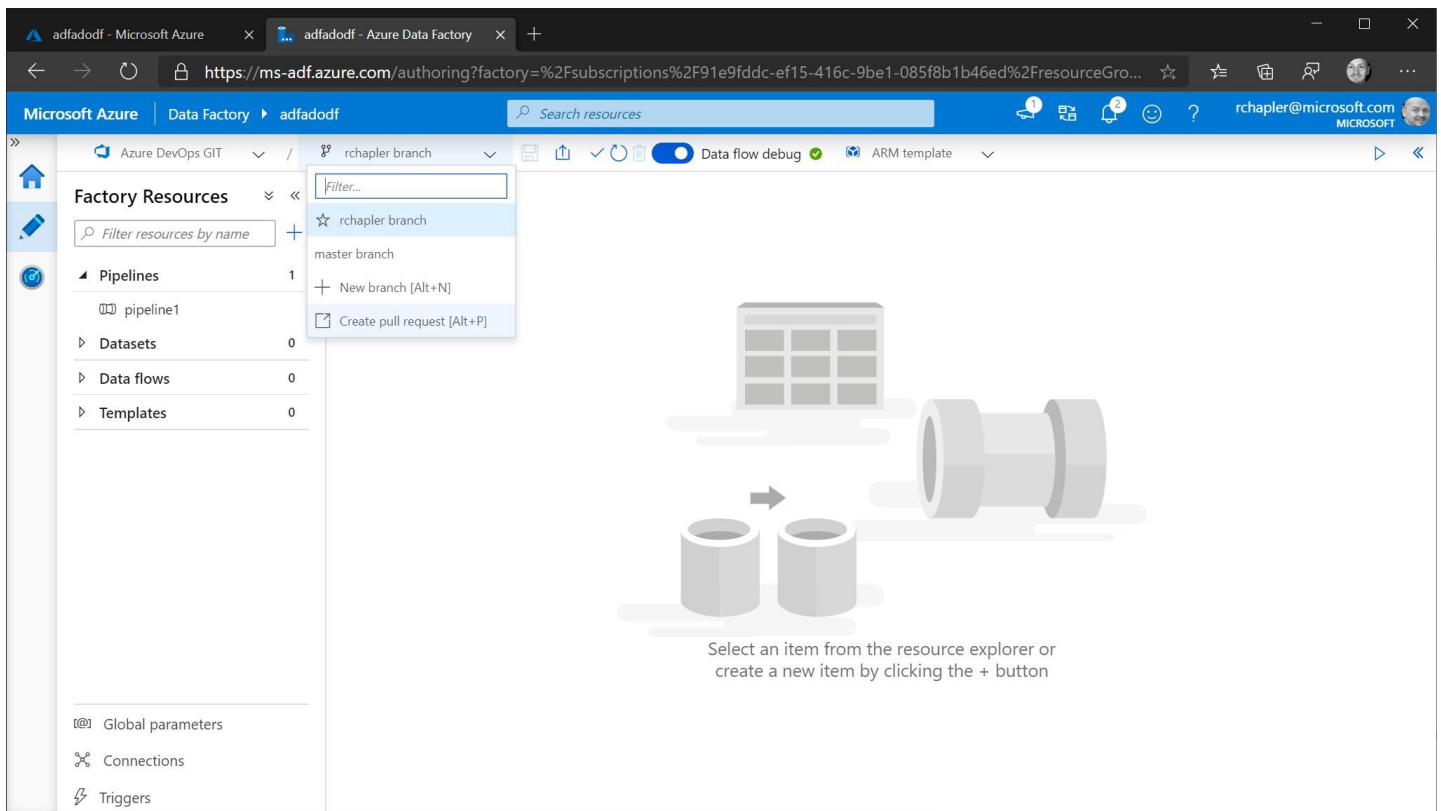
NAME	TYPE	RUN START	DURATION	STATUS	INTEGRATION I
Set variable3	SetVariable	2020-04-06T19:22:02.836	00:00:00	Failed	
If Condition1	IfCondition	2020-04-06T19:22:02.430	00:00:01	Failed	Unknown
Set variable1	SetVariable	2020-04-06T19:22:01.992	00:00:01	Succeeded	Unknown

Setting the Flag variable to False will force a pipeline failure (i.e. don't worry about the Status “Failed” message... that is what it is supposed to do).

Click the “Save” button. Note that save will capture a local copy but doesn't imply a publication or deployment to the repository.

Pull Request

Next, we will “promote” changes from the working branch to the master branch via Pull Request.



Click on the branch dropdown control and select “Create pull request (Alt+P)” from the resulting dropdown.

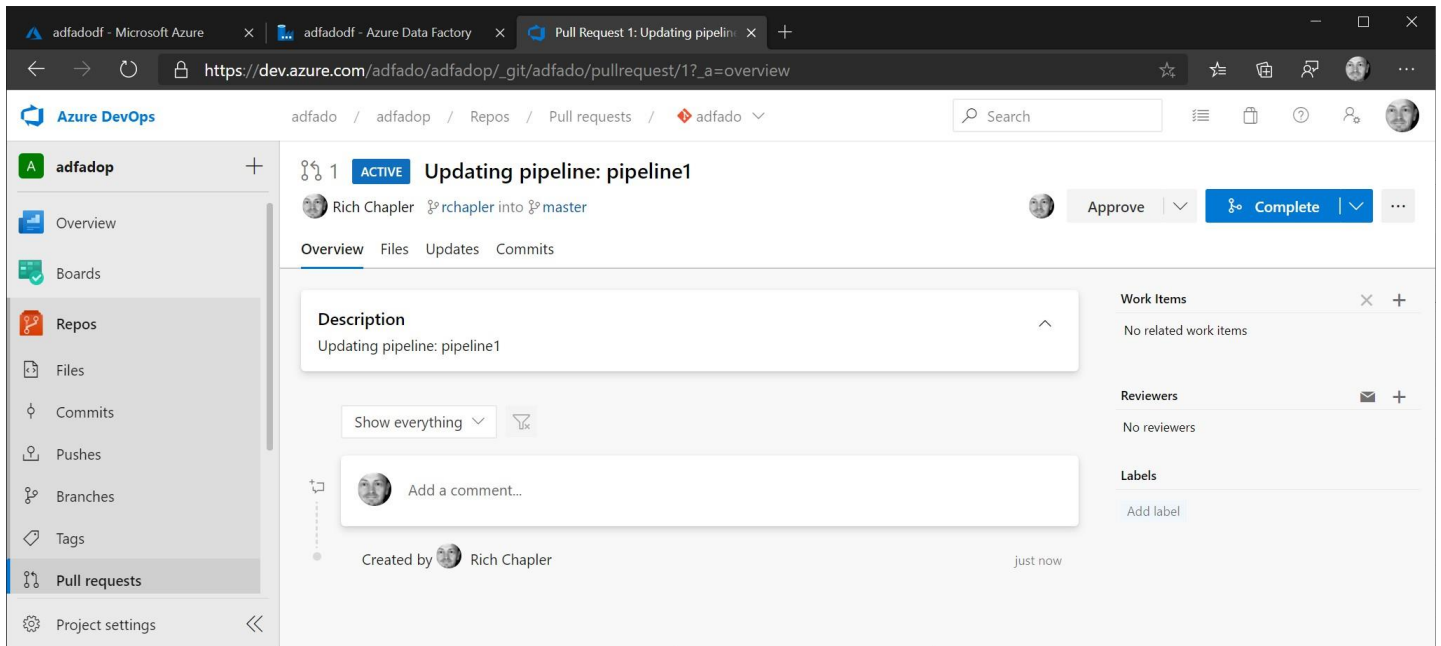
A third tab (pointing to <https://dev.azure.com...>) will open.

The screenshot shows the 'New Pull Request' interface in Azure DevOps. The left sidebar contains navigation links for Overview, Boards, Repos, Pull requests (selected), Pipelines, Test Plans, Artifacts, and Compliance. The main area is titled 'New Pull Request' and includes fields for source and target branches (rchapter into master), a title, a description, reviewers, and work items. A diff view at the bottom shows a change in 'pipeline1.json' where the 'value' of a 'Flag' variable is being updated from true to false. The 'Create' button is located at the bottom right of the form.

On the “New Pull Request” page, review and / or populate following:

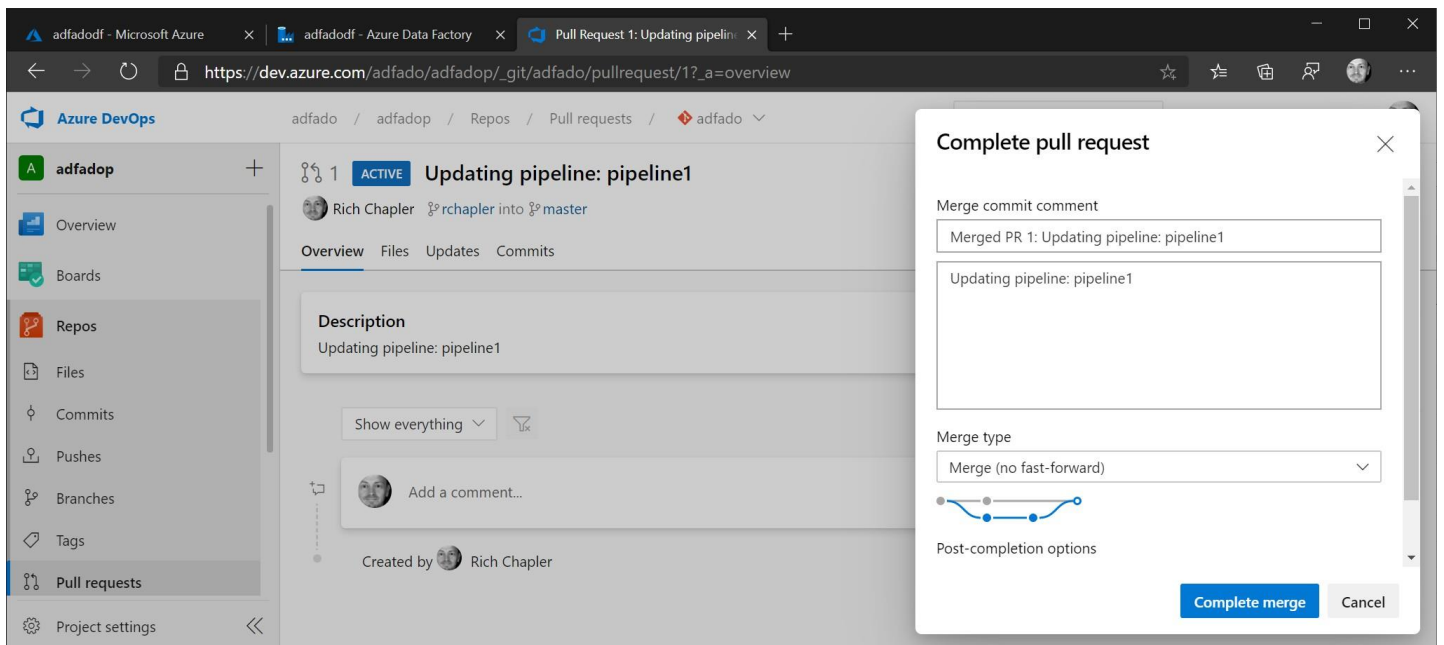
- {branch name} into master ... reflects that intent of the pull request (i.e. promote from branch to master)
- Title ... confirm default or create a new title that is meaningful for you (and aligned with your naming standards)
- Description ... confirm default or create a new description that is meaningful for you and your team
- Reviewers ... enter the alias(es) for user(s) or group(s) that are responsible for reviewing / approving code changes
- Work Items ... select Work Items related to the code changes related to your current assignment
- Files / Commits ... review the detailed changes to files / code and address conflicts (if applicable)

Click the “Create” button. Allow time for processing.

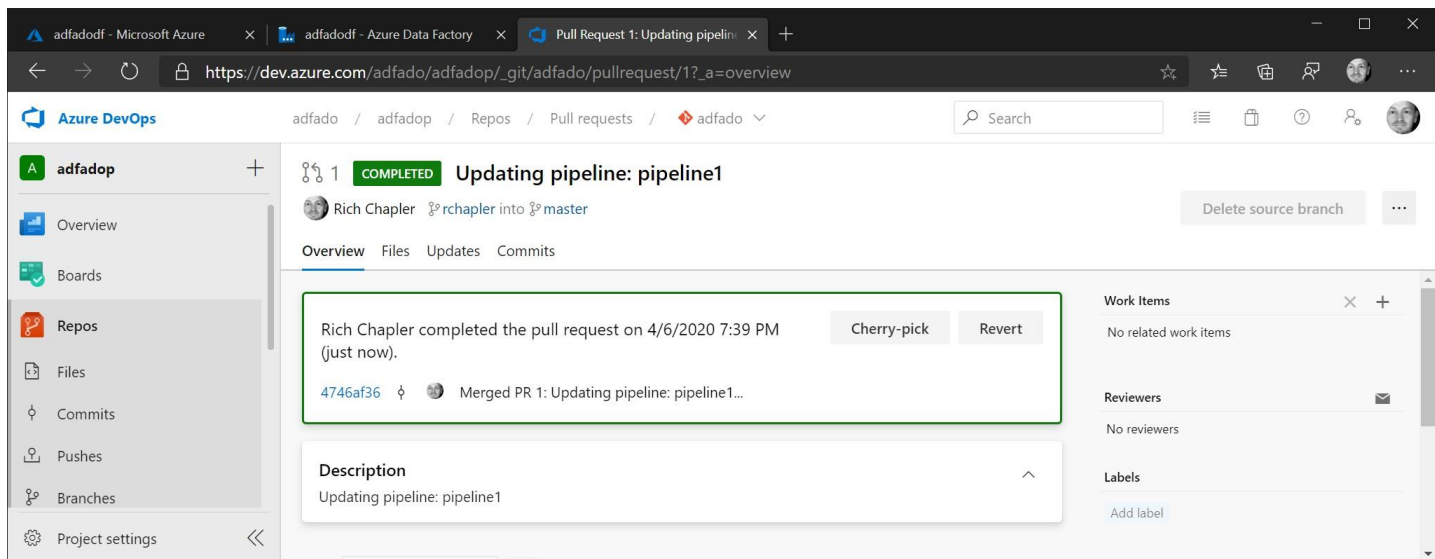


At this point, designated reviewers will be notified by email and they will need to review and approve for the Pull Request to move forward. Additional configuration can provide for validation of other gating criteria {e.g. inclusion of text patterns that might be secret like an account key}.

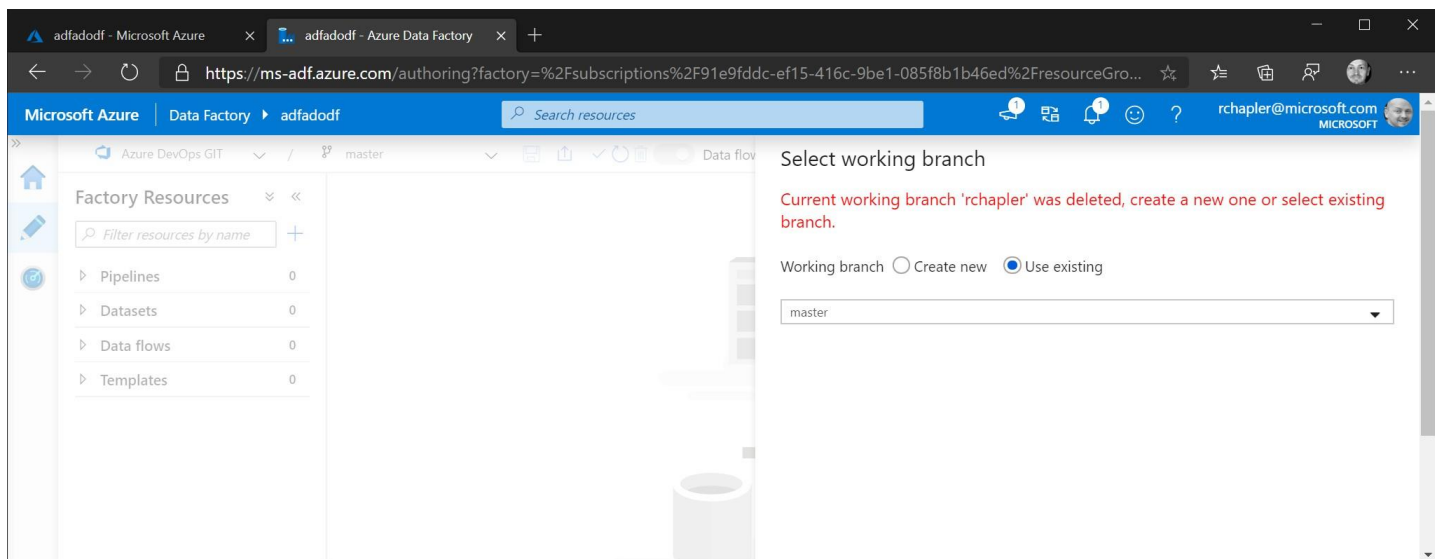
We haven't included gating factors in this example. Click the "Complete" button.



Update the fields in the "Complete pull request" popout. Click the "Complete merge" button. Allow time for processing.



Note post-completion options such as “Revert” that provide for operational control even after a code merge. Return to your data factory (page refresh if you’re simply switching tabs).



Note that, once again, the data factory is asking you to select an existing or create a new working branch.

