# Lab: Automate business processes with Logic Apps

# Student lab manual

## Lab scenario

Your organization keeps a collection of JSON files that it uses to configure third-party products in a Server Message Block (SMB) file share in Microsoft Azure. As part of a regular auditing practice, the operations team would like to call a simple HTTP endpoint and retrieve the current list of configuration files. You have decided to implement this functionality using a no-code solution based on Azure API Management service and Logic Apps.

## Objectives

After you complete this lab, you will be able to:

* Create a Logic App workflow.
* Manage products and APIs in a Logic App.
* Use Azure API Management as a proxy for a Logic App.

## Lab setup

* Estimated time: **45 minutes**

## Instructions

### Before you start

#### Sign in to the lab virtual machine

Ensure that you're signed in to your Windows 10 virtual machine (VM) by using the following credentials:

* Username: **Admin**
* Password: **Pa55w.rd**

#### Review the installed applications

Find the taskbar on your Windows 10 desktop. The taskbar contains the icons for the applications that you'll use in this lab:

* Microsoft Edge
* File Explorer
* Windows Terminal

### Exercise 1: Create Azure resources

#### Task 1: Open the Azure portal

1. Sign in to the Azure portal (<https://portal.azure.com>).
2. If this is your first time signing in to the Azure portal, you'll notice a dialog box offering a tour of the portal. Select **Get Started** to skip the tour.

#### Task 2: Create an API Management resource

1. In the Azure portal, create a new API Management account with the following details:
   * New resource group: **AutomatedWorkflow**
   * Name: **prodapim*[yourname]***
   * Location: **East US**
   * Organization name: **Contoso**
   * Pricing tier: **Consumption (99.9 SLA, %)**

* **Note**: Wait for Azure to finish creating the API Management resource prior to moving on in the lab. You will receive a notification when the resource is created.

#### Task 3: Create a Logic App resource

1. In the Azure portal, create a new **logic app** with the following details:
   * Existing resource group: **AutomatedWorkflow**
   * Name: **prodflow*[yourname]***
   * Location: **Region - East US**
   * Log Analytics: **Off**

* **Note**: Wait for Azure to finish creating the Logic Apps resource prior to moving on in the lab. You will receive a notification when the resource is created.

#### Task 4: Create a storage account

1. Create a new storage account with the following details:
   * Existing resource group: **AutomatedWorkflow**
   * Name: **prodstor*[yourname]***
   * Location: **(US) East US**
   * Performance: **Standard**
   * Account kind: **StorageV2 (general purpose v2)**
   * Replication: **Locally-redundant storage (LRS)**
   * Access tier: **Hot**

* **Note**: Wait for Azure to finish creating the storage account before you move on in the lab. You'll receive a notification when the account is created.

#### Task 5: Upload sample content to Azure Files

1. Access the **prodstor*[yourname]*** storage account that you created earlier in this lab.
2. Select the **File shares** link in the **File service** section, and then create a new share with the following settings:
   * Name: **metadata**
   * Quota: **1** (GiB)
3. Select the recently created **metadata** share.
4. In the **metadata** share, select **Upload** to upload the following files in the **Allfiles (F): \Allfiles\Labs\09\Starter** folder on your lab VM.
   * **item\_00.json**
   * **item\_01.json**
   * **item\_02.json**
   * **item\_03.json**
   * **item\_04.json**

* **Note**: We recommend that you enable the **Overwrite if files already exist** option.

#### Review

In this exercise, you created all the resources that you'll use for this lab.

### Exercise 2: Implement a workflow using Logic Apps

#### Task 1: Create a trigger for the workflow

1. Access the **prodflow*[yourname]*** logic app that you created earlier in this lab.
2. On the **Logic Apps Designer** blade, select the **Blank Logic App** template.
3. In the **Designer** area, add a new **When a HTTP request is received (Request)** trigger with the following details:
   * Method: **GET**

#### Task 2: Create an action to query Azure Storage file shares

1. In the **Designer** area, add a new **List files (Azure File Storage)** action with the following details:
   * Connection Name: **filesConnection**
   * Storage Account: **prodstor*[yourname]***
   * Folder: **/metadata**

#### Task 3: Create an action to project list item properties

1. In the **Designer** area, add a new **Select (Data Operations)** action with the following details:
   * From: **value (List files)**
   * Map mode: **Text**
   * Map: **Name (List files)**

#### Task 4: Build an HTTP response action

1. In the **Designer** area, add a new **Response (Request)** action with the following details:
   * Status Code: **200**
   * Body: **Output (Select)**
2. **Save** the workflow.

#### Review

In this exercise, you built a basic workflow that starts when it's triggered by an HTTP GET request. It then queries a storage service, enumerates the results, and then returns those results as an HTTP response.

### Exercise 3: Use Azure API Management as a proxy for Logic Apps

#### Task 1: Create a new product

1. Access the **prodapim*[yourname]*** API Management resource that you created earlier in this lab.
2. Navigate to the empty list of **Products** for the new account.

* **Note**: Consumption accounts start without any products.

1. Add a new product with the following details:
   * Display name: **Free**
   * Id: **free**
   * Description: **Free tier for anonymous use**
   * State: **Published**
   * Requires subscription: **false**

#### Task 2: Create an API integrated with Logic Apps

1. Navigate to the empty list of **APIs** for the account.
2. Create a new **Logic App**–integrated API with the following details:
   * Logic App: **prodflow*[yourname]***
   * Display name: **Metadata Lookup**
   * Name: **metadata-lookup**
   * Description: **Look up metadata JSON files**
   * Products: **Free**

#### Task 3: Test the API operation

1. Navigate to the **Test** tab for the new API.
2. On the **Test** tab, perform the following actions:
   1. Select the single **GET** operation.
   2. Copy the value of the **Request URL** field. (You will use this value later in the lab.)
   3. Perform a test request against the operation.
   4. Observe the JSON results of the test request.

#### Task 4: Validate the end-to-end solution using httprepl

1. Open the **Windows Terminal** application.
2. Start the **httprepl** tool, and set the base Uniform Resource Identifier (URI) to the value of the **Request URL** for the API operation that you copied earlier in this lab using the following example:

* httprepl <api-operation-request-url>
* **Note**: For example, if your **URL** is **https://prodapimstudent.azure-api.net/manual/paths/invoke**, your command would be **httprepl https://prodapimstudent.azure-api.net/manual/paths/invoke**.

1. Within the tool prompt, run the **get** command against the API endpoint:

* get

1. Observe the JSON response content of the function app response.
2. Exit the **httprepl** application:

* exit

1. Close the currently running Windows Terminal application.
2. Return to your browser window within Azure portal.

#### Review

In this exercise, you used Azure API Management as a proxy to trigger your Logic App workflow.

### Exercise 4: Clean up your subscription

#### Task 1: Open Azure Cloud Shell and list resource groups

1. In the Azure portal, select the **Cloud Shell** icon to open a new shell instance.
2. In the Azure portal, at the **Cloud Shell** command prompt, enter the following command, and then select Enter to list all resource groups in the subscription:

* az group list

1. Enter the following command, and then select Enter to get a list of possible commands to delete a resource group:

* az group delete --help

#### Task 2: Delete resource groups

1. Enter the following command, and then select Enter to delete the **AutomatedWorkflow** resource group:

* az group delete --name AutomatedWorkflow --no-wait --yes

1. Close the Cloud Shell pane.

#### Task 3: Close the active applications

* Close the currently running Microsoft Edge application.

#### Review

In this exercise, you cleaned up your subscription by removing the resource groups that were used in this lab.

©2019 Microsoft Corporation. All rights reserved. The text in this document is available under the [Creative Commons Attribution 3.0 License](https://creativecommons.org/licenses/by/3.0/legalcode), additional terms may apply. All other content contained in this document (including, without limitation, trademarks, logos, images, etc.) are ***not*** included within the Creative Commons license grant. This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it. Some examples are for illustration only and are fictitious. No real association is intended or inferred. Microsoft makes no warranties, express or implied, with respect to the information provided here.