**Enterprise Integration with Logic Apps**

Student Lab Manual

V1.0, July 28, 2019

Information in this document is subject to change without notice. The example companies, organizations, products, people, and events depicted herein are fictitious. No association with any real company, organization, product, person or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarked, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2019 Microsoft Corporation. All rights reserved.

Microsoft, MS-DOS, MS, Windows, Windows NT, MSDN, Active Directory, BizTalk, SQL Server, SharePoint, Outlook, PowerPoint, FrontPage, Visual Basic, Visual C++, Visual J++, Visual InterDev, Visual SourceSafe, Visual C#, Visual J#, and Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the U.S.A. and/or other countries.

Other product and company names herein may be the trademarks of their respective owners.

Contents

[Enterprise Integration with Logic Apps 4](#_Toc15232144)

[Introduction 4](#_Toc15232145)

[Objectives 4](#_Toc15232146)

[Prerequisites 4](#_Toc15232147)

[Exercise 1: Creating your Integration account 5](#_Toc15232148)

[Scenario 5](#_Toc15232149)

[Task 1 – Create your integration account 5](#_Toc15232150)

[Task 2 – Add your schemas and map 7](#_Toc15232151)

[Exercise 2: Creating your Logic App 10](#_Toc15232152)

[Scenario 10](#_Toc15232153)

[Task 1 – Create your Logic App 10](#_Toc15232154)

[Task 2 – Testing your integration Logic App 18](#_Toc15232155)

# Enterprise Integration with Logic Apps

## Introduction

Estimated time to complete this lab

60 minutes

#### Overview

For business-to-business (B2B) workflows and seamless communication with Azure Logic Apps, you can create an integration account that represents an enterprise organization. Organizations can exchange messages through industry-standard protocols, including AS2, X12, and EDIFACT. You can also secure messages with both encryption and digital signatures.

Integration accounts are cloud-based containers that store all your artifacts, like schemas, partners, certificates, maps, and agreements. You can use these artifacts to design, deploy, and maintain your B2B apps and also to build B2B workflows for logic apps. But before you can use these artifacts, you must first link your integration account to your logic app. After that, your logic app can access your integration account's artifacts.

### Objectives

In this hands-on lab, you will learn how to:

* Create an Azure Integration account
* Add pre-existing artifacts to your integration account
* Use logic app actions to receive, decode, validate and map flat-file and XML documents

### Prerequisites

The following is required to complete this hands-on lab:

* Microsoft V[isual Studio 2017 Professional or Enterprise edition](http://www.microsoft.com/visualstudio/), version 15.9.13+
* [Microsoft Azure SDK for .NET for Visual Studio 2017](http://www.microsoft.com/windowsazure/sdk/)
* A Microsoft Azure subscription

# Exercise 1: Creating your Integration account

### Scenario

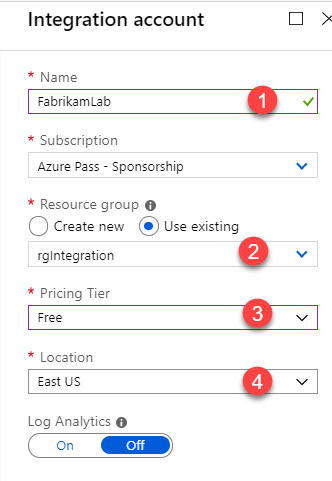
Enterprise Integration accounts combined with logic apps enables business integration using a declarative programming model with message patterns such as VETER (Validate-Enlarge-Transform-Enlarge-Route) for a multi-tenant cloud environment. Messages received in the VETER process flows to the destination partner/location via the logic app.

Logic apps are used as a bridge between the source and destination endpoints, where the inbound message can be targeted at one or multiple endpoints.

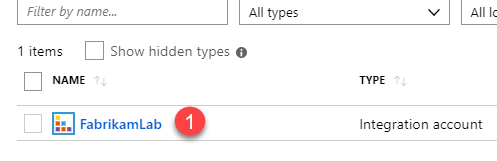
A very popular process in enterprises is the translation and validation of documents that are in one format, to documents in another format. In this lab, you’ll be transforming a *Buyers* flat file into a *Customers* XML document. This same process you will build into your logic app could also transform other document formats, all you would need to do is build your schemas and associated maps and store those in your integration account.

## Task 1 – Create your integration account

1. Log into the Azure portal create a new resource group to contain your integration account.
2. In your resource group blade, choose the **+Add** toolbar button and type ‘*Integration account*’ into the marketplace search field.
3. Click the **Create** button on Integration Account blade.
4. You can name the integration account anything you wish, usually it would be a partner/company name, such as Fabrikam etc. Make sure you choose the Free pricing tier and put your integration account in the same region where your logic app will be located. Click the **Create** button.

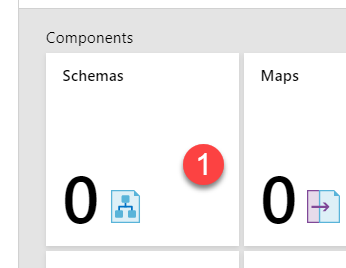


1. Once your integration account has been created, click on its icon in the resource group blade.

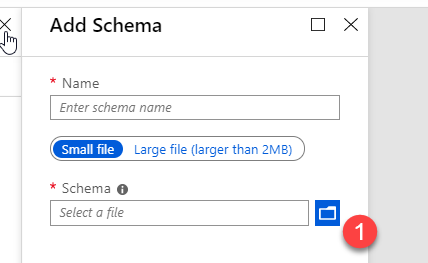


## Task 2 – Add your schemas and map

1. The first thing you will do is add the two schemas that will be used for the lab, a flat file schema and an XML schema. Click on the Schemas tile to open the schemas blade.



1. Click on the **+Add** toolbar button. Here, you do not need to enter a schema name unless you want to, when you select the schema file, the name will automatically be the name of the file. Choose the file icon in the **Add Schema** blade.



1. Browse to the **.\LogicApps\Labs\Integration\Assets** folder and select the Customers.xsd file. Select the Ok button on the Add Schema blade.

The Customers.xsd schema will represent and xml document that looks similar to this:  
**<?xml version="1.0" encoding="UTF-8"?>**

**<Customers>**

**<Customer>**

**<FirstName>John</FirstName>**

**<LastName>Doe</LastName>**

**<Address>One Microsoft Way</Address>**

**<City>Redmond</City>**

**<State>WA</State>**

**<Zip>98052</Zip>**

**</Customer>**

**</Customers>**

A sample Customers.xml document can be found in the Assets folder.

1. Once again, click on the **+Add** toolbar button. This time browse to the **Assets** folder and select the **BuyerFlatFileSchema2.xsd** file. Select **Ok** on the Add Schema blade.

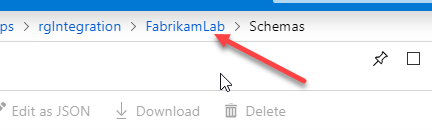
The BuyerFlatFileSchema2 schema will represent a flat-file document that looks similar to this:

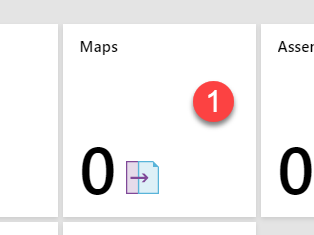
**John,Doe,One Microsoft Way,Redmond,WA,98052**

**John,Wayne,123 Hollywood Ct.,Hollywood,CA,12345**

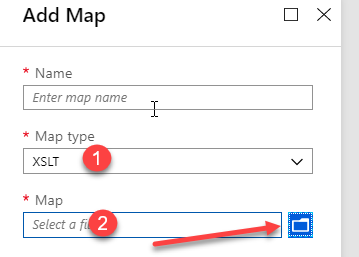
**George,Patton,456 Patton Rw.,Phoenix,AZ,43234**

A sample **Buyers.txt** file can be found in the Assets folder.

1. Go back to the Overview blade of your integration account be selecting the name of your integration account on the breadcrumb link:  
     
   
2. In order to transform the flat file into the XML document, you will need a map. A map is typically an XSLT template that describes how one field of data maps to an associated field in the transformed document. A map will also describe data types, repeating fields and other formatting details. Click on the Maps tile.

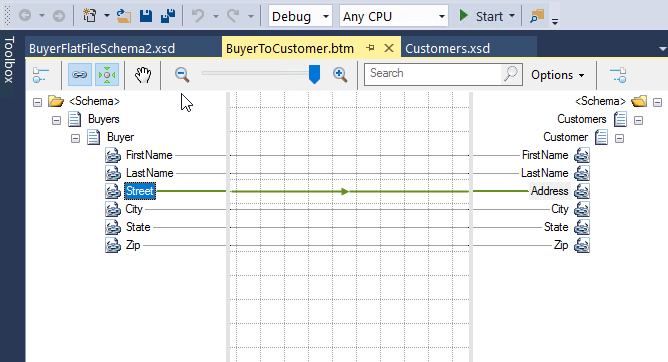


1. Click the **+Add** toolbar button and then select the file icon on the **Add Map** blade. Leave the Map type set to XSLT.



1. Browse to and select the **.\LogicApps\Labs\Integration\Assets\BuyerToCustomer.xsl** file (you may have to change the file type in the Open dialog box to see the file name). Select Ok on the Add Map blade and then go back to your integration accounts Overview blade.

The *BuyerToCustomer.xsl* map was created with the Microsoft BizTalk mapper tool (along with the schemas). The map represents a simple mapping where you map multiple buyers to multiple customers. In reality, you’re probably going to find that creating technically accurate schemas and maps is more difficult than working with integration accounts and logic apps, regardless of which tools you use!



Your integration account is now complete!

# Exercise 2: Creating your Logic App

### Scenario

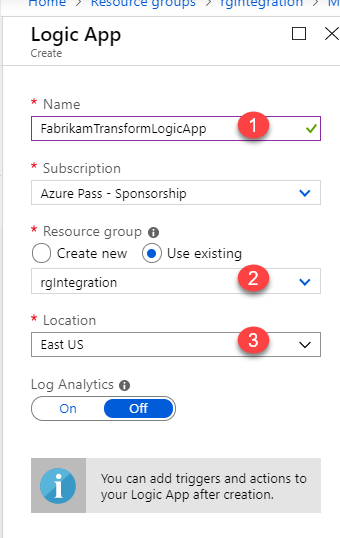
Your Integration Account is now ready to go with the artifacts required for your logic app.

The logic app you will be creating will perform the following steps:

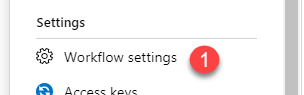
* It will have an http trigger that will accept the incoming flat-file data
* The data coming in from the http trigger will be decoded
* The decoded data will be validated with an XML validator
* An xpath expression will be applied to the validated XML
* Finally, if everything is good up to this point, the transformation will take places to map the data to the XML document

## Task 1 – Create your Logic App

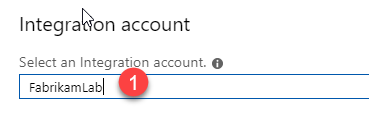
1. Within you resource group where the integration account is, select the +Add toolbar button and enter Logic app. Click on the Create button.
2. Give your logic app a name and make sure it’s in the same Location as your integration account. Click the Create button.



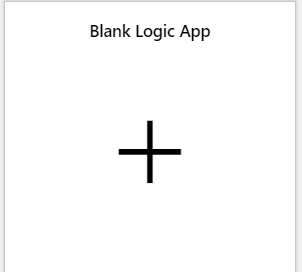
1. Once the logic app has been created, you need to link your logic app to your integration account. Within your resource group, click on the name of your logic app to advance to the logic app blade. If the portal advances you to the Logic App designer, go back to the Overview blade instead.
2. Select the Workflow settings menu item.



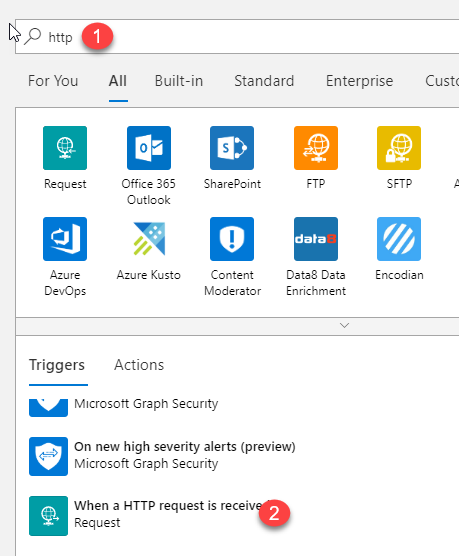
1. From the Integration account drop-down, choose your integration account name. Choose the **Save** button on the toolbar.



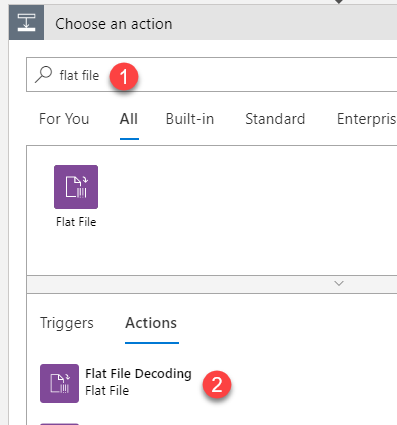
1. Click on the **Logic app designer** menu item.
2. Choose the **Blank Logic App** template tile.



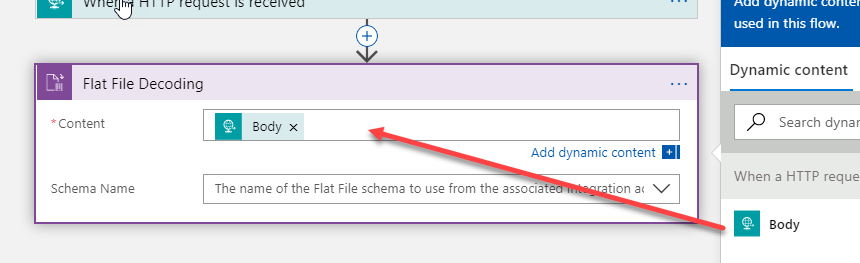
1. In the logic app design window, type in ‘*http’* and then scroll down and choose the ‘**When HTTP request is received’** icon. Unlike in the last lab, you will not need to choose a JSON schema for the data being passed in, you will be passing in just flat-file data. Save the logic app.



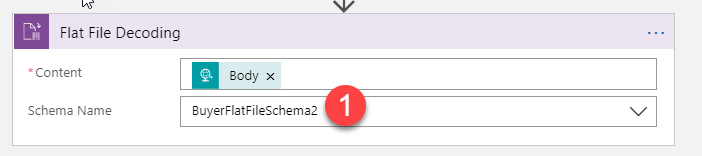
1. Click the **+New step** button.
2. Type in ‘*flat file’* and you should see the **Flat File Decoding** action. Select this action. The flat file decoder action outputs an xml document that you may do further conversions on. It also helps validate that the incoming flat file structure matches the schema.



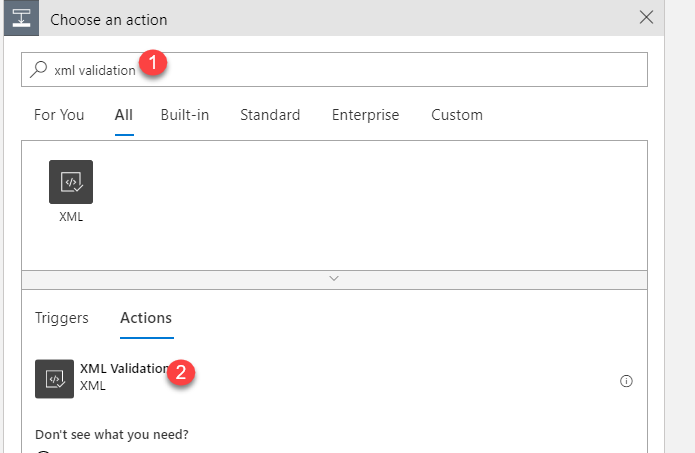
1. For the **Content**, click your mouse in the *content* edit field and on the right-hand side, choose ‘**See more’**. From there, click on **Body**. This will insert the Body into the content field.



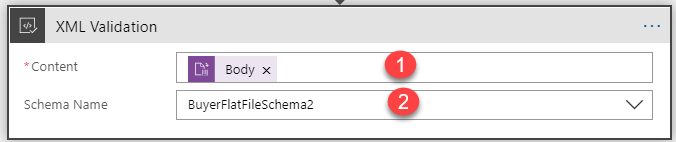
1. Click on the **Schema Name** drop-down and select the **BuyerFlatFileSchema2** schema. Save the logic app.



1. Click on the **+New step** button.
2. Type ‘*xml validation’* into the search field and then select the **XML Validation** action. In the previous step, the flat file decoding step decoded the flat file document in to XML, now you need to validate that XML against the SAME schema as in the last step. This may seem like an unnecessary step, but it is the kind of guarantee that most B2B processes expect as a document flows through the system.



1. For the Content, click in the content edit field and then choose **Body** in the dynamic content window.
2. For the Schema Name, choose the **BuyerFlatFileSchema2** schema. Save the logic app.

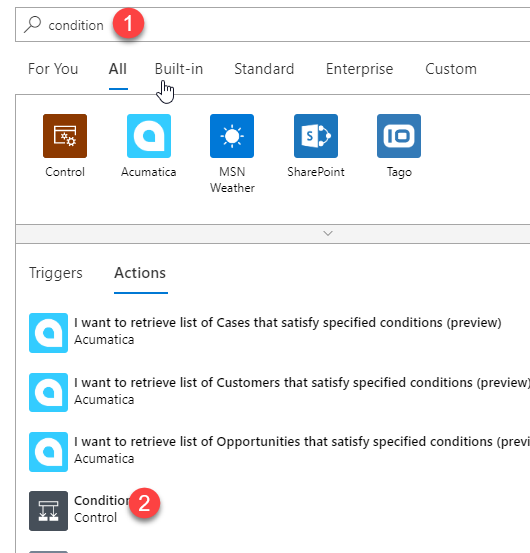


1. Now, you have gotten the flat file in to the process, decoded it to XML and validated the XML. The next thing that typically happens is that you need to confirm that the incoming XML document only has one root node. You cannot map one document format to another with the provided map if the document has more than one root node.

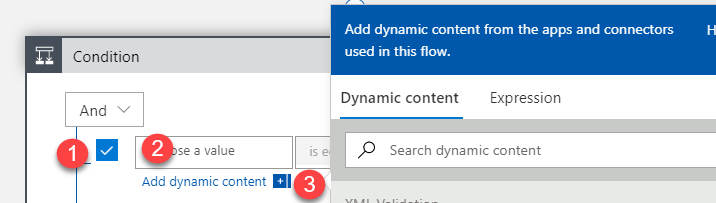
To validate this, you will using an xpath expression.

Click on the **+New step** button.

1. Type in **Condition** into the search field and select the **Condition** control icon.

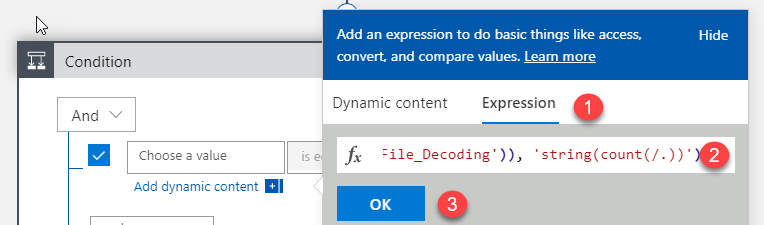


1. Click the checkbox adjacent to the first edit field and place your cursor in that field. Select the **Add dynamic content** link if the dynamic content box does not appear.



1. Click on **Expression** in the dynamic content field. Either paste in or type the text below and then select the OK button.

**xpath(xml(body(**'Flat\_File\_Decoding'**))**,'string(count(/.))'**)**



1. Leave the middle field set to ‘is equal to’ and put your cursor in the last edit field on the right-hand side. Go through the same process as before, selecting dynamic content -> Expression and then put in the text:  
   **string(‘1’)**
2. Save the logic app.
3. Before going any further, let’s look at the full expression that you just created to understand what is going on. Click on the **</> Logic app code review** menu item.

In the code you will see this:  
  
 "equals": [

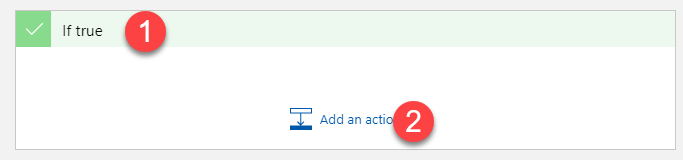
"@xpath(xml(body('Flat\_File\_Decoding')), 'string(count(/.))')",

"@string('1')"

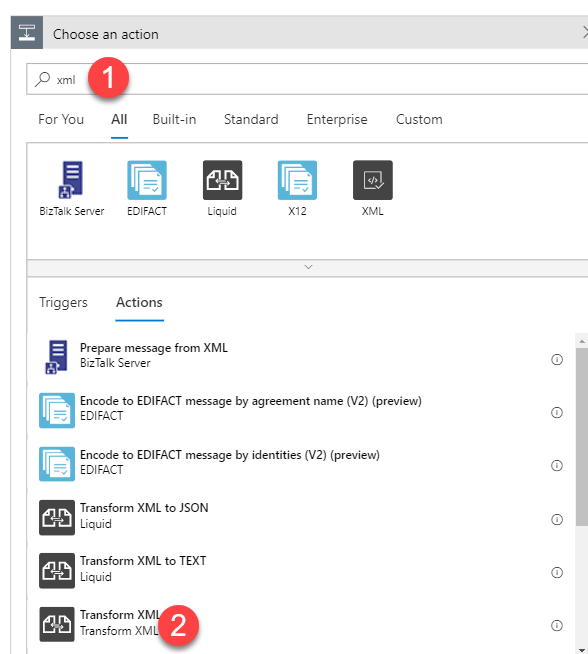
* The first part of the expression highlighted above, takes the output of the flat file decoding action and makes sure the body represents an XML document
* Next, the Xpath code says ‘get a count of the number of root nodes ‘/’ starting with the first root node as the current node ‘.’. All this of course has been turned in to a string by XPath.
* Finally, this output is compared to the string ‘1’ to make sure there is only one root node.

Go back to the Logic App designer.

1. Click on the **Add an action** link in the **If true** branch.



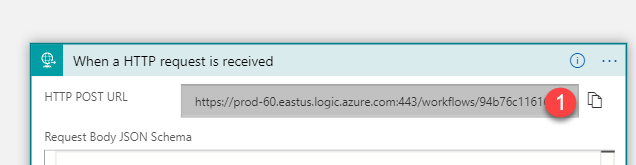
1. In the search field type in ‘**xml’** and then look for the **Transform XML** icon. Click on the icon.



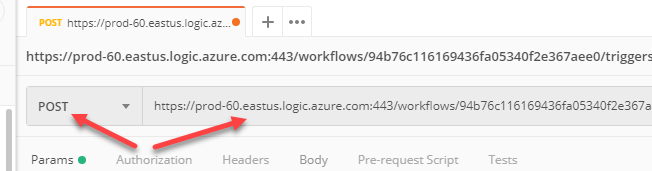
1. Put your cursor in the **Content** field and select **Body** from the Flat File Decoding section of the dynamic content dialog. For the **Map**, click on the Map drop-down control and select the **BuyerToCustomer** map. This is the step in the workflow that actually does the full transformation from flat file to XML.
2. Save the logic app.

## Task 2 – Testing your integration Logic App

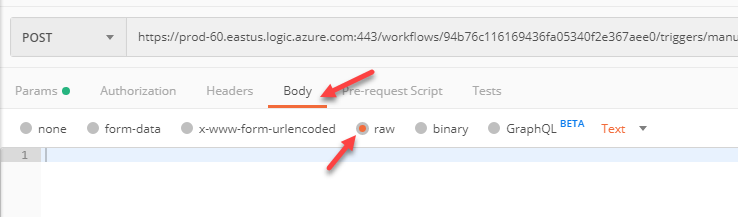
1. Scroll to the top of your logic app and click on the Http request action. You should notice that you have a full URL available to you. Copy this URL to your clipboard.



1. Open Postman or tool of your choice that allows you to perform POST commands with data to your logic app.
2. Select a new tab in Postman and paste your URL into the address field. Make sure you select **POST** as the action you’re going to perform.

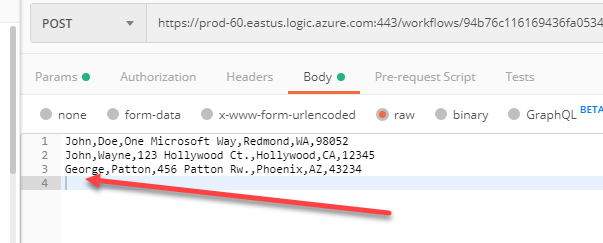


1. You do not need to modify the header settings because you will be putting in plain text. Click on the **Body** menu item and then the ‘**raw’** choice button.

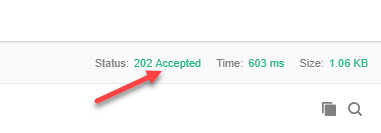


1. You can either put in your own records or you can open up the **Buyers.txt** file located in the **.\LogicApps\Labs\Integration\Assets** folder. Copy the contents of the Buyers.txt into the Body field.

Make sure that at the end of the last line of text, you select Enter to make sure that you have a carriage return/line feed at that location.



1. Within Postman, click the Send button. What you should see is a 202 Accepted response in Postman.



1. Go back into the portal to the **Overview** blade for your logic app. Click on the **Refresh** toolbar button and then look at the Status in the **Runs history** section.
2. Click on the status and you will be taken to the ‘Logic app run’ blade where you can see the steps that your logic app took to complete the workflow. You can click on any of the actions and the action will expand and show you the details.

