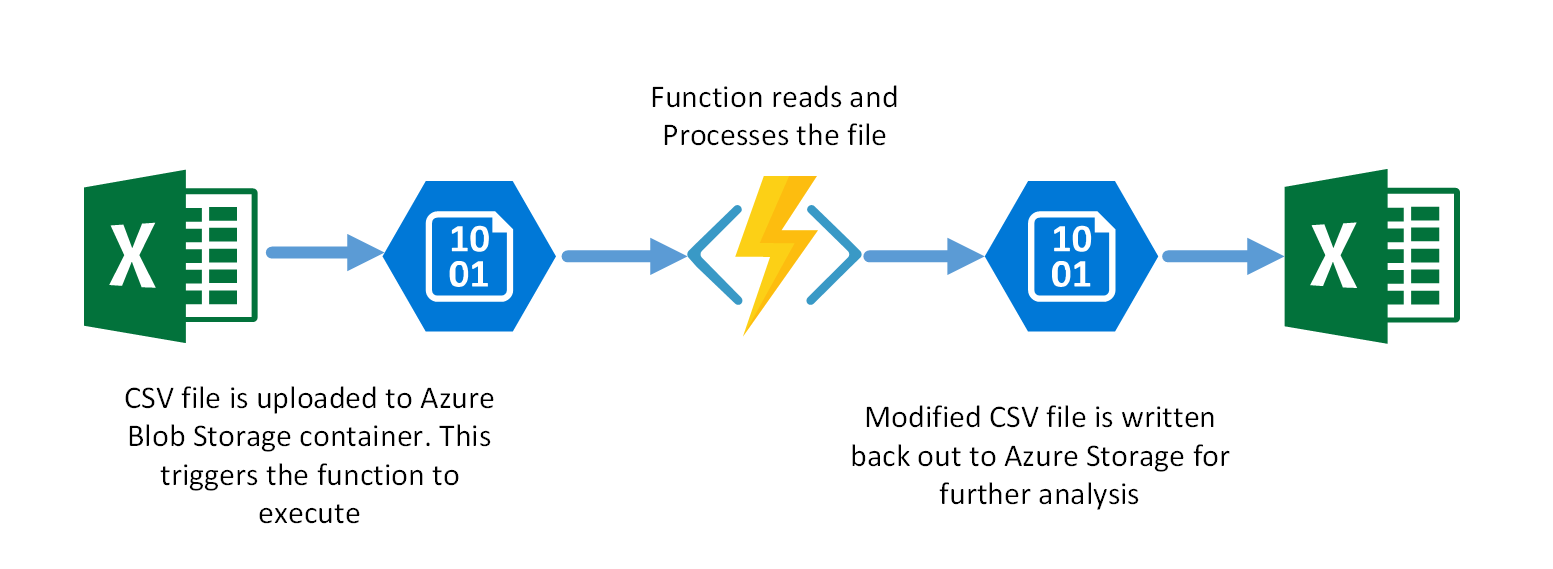
Most of this lab has been referenced from <https://azurecitadel.github.io/labs/functions/#create-the-function-app> with changes/updates as needed.

**UseCase:**

The “usecase” we want to solve as the subject of this lab is that we have a csv file containing the details of survivors of the 1914 Titanic Disaster in which the passenger name column is in the format “surname, [title] forename(s) [, suffix]” and we need to reformat it to “title forenames surname suffix” and store the result in a new column for later display. There are also some name entries which do not confirm to this layout and we need to highlight them so that they can be fixed-up. We’ll use a simple function app to do this.

There are many ways to create and deploy a function app, but in this lab we’re going to create a simple C# function app through the Azure Portal. This function will be triggered by the arrival of a new csv file in a defined Azure Storage Container. The function will read the csv file, process it and write-out a new csv file to another blob.

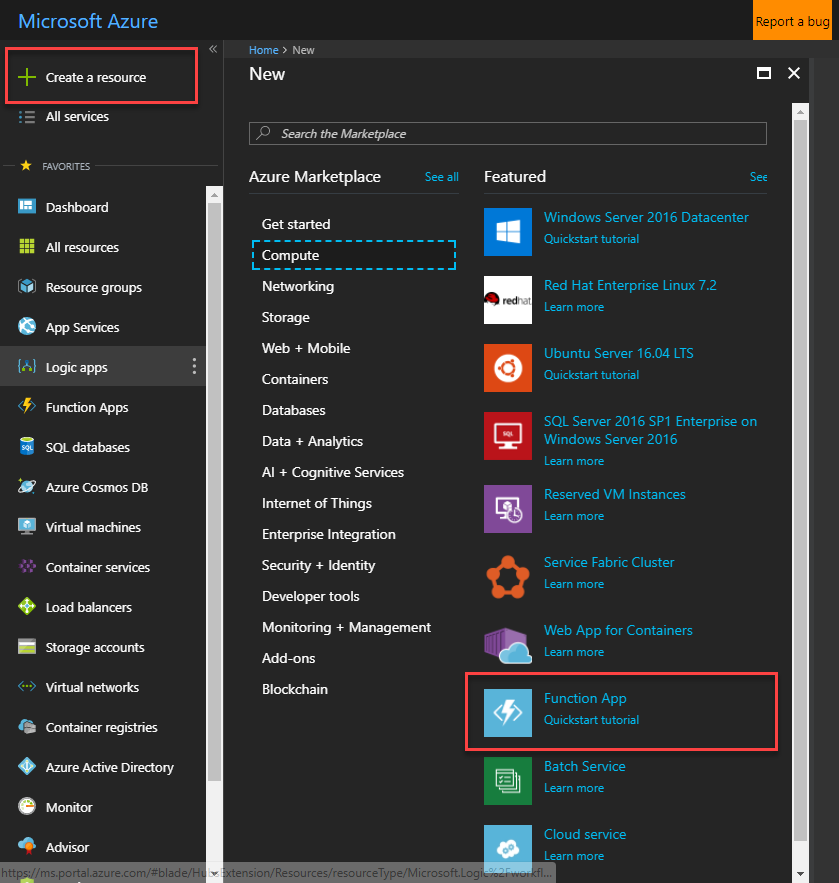


**Pre-requisites**

* Azure Subscription
* Basic knowledge of C#

## Create the function app

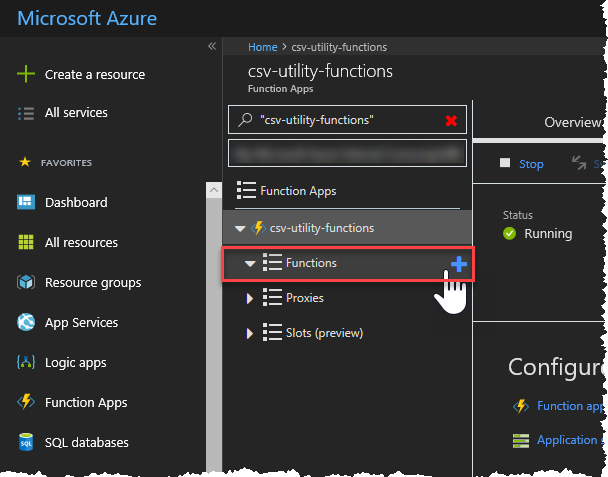
* Login to the [Azure Portal](https://portal.azure.com/)
* Click the **Create a Resource** button and select **Compute**. Then select **Function App**.



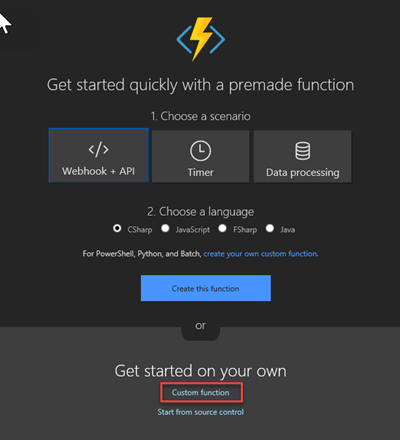
* Set the following settings for the app:
  + App name: **csv-utility-functions-[yourname]** (ensure this is unique)
  + Resource Group: **csv-utility-functions-rg**
  + OS: **Windows**
  + Hosting Plan: **Consumption Plan**
  + Location: **East US** (or as desired)
  + Storage: **Create new** and enter a unique name
  + Application Insights: **Off**
* Click **Create**. This will deploy your function app. Once this has completed, navigate to it.

Now that we have a function app to host our function, we can create the function itself.

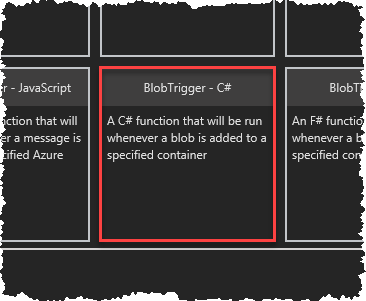
* Expand the function app, then click the **+** next to **Functions** to create a new function



* Click the **Custom Function** link at the bottom of the page



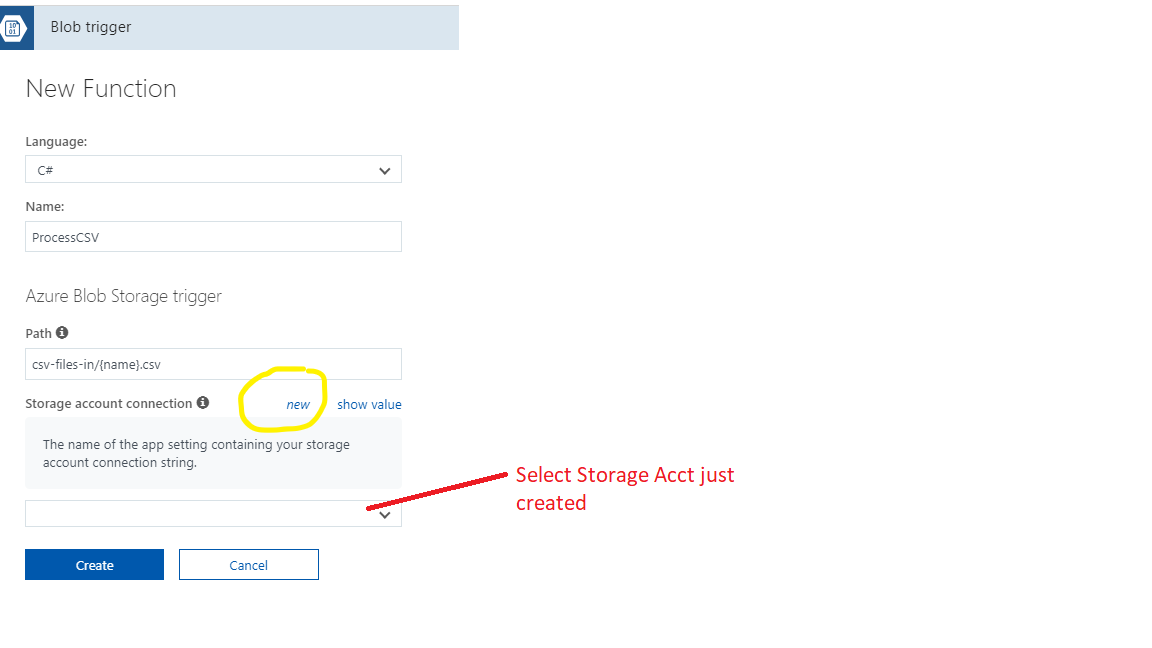
* Since we want to trigger our function based on the arrival of a new blob, select the **BlobTrigger - C#**template.



* Set the following Azure Blob Storage trigger settings:

| **Setting** | **Value** | **Description** |
| --- | --- | --- |
| **Name of your function** | ProcessCSVFile | The function name |
| **Path** | csv-files-in/{name}.csv | This is the name of the storage container where csv files t |

Create a “new” storage account by clicking on “new” 🡪 In the following blade, click on “Create New”, and give the storage account a unique name.

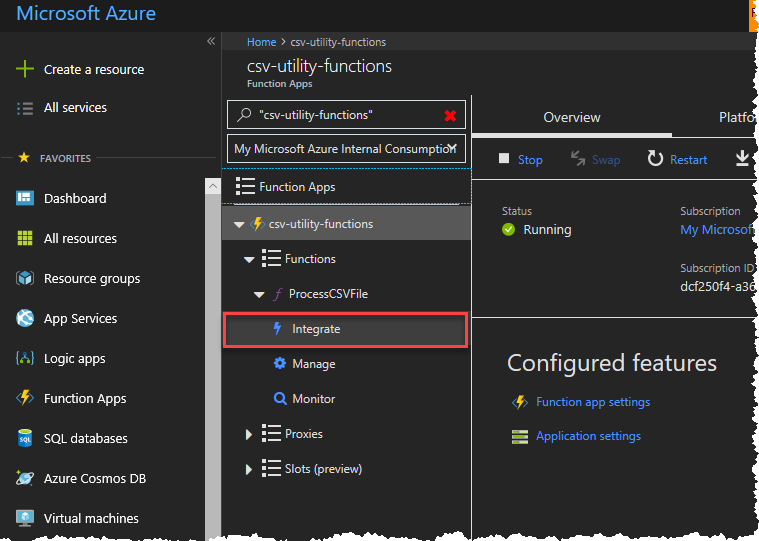
****

* Click **Create** to create the new function.

### Define the output blob container

Next we need to define a blob container in our storage account that we will use to store the processed csv files.

* Expand the function and click **Integrate**

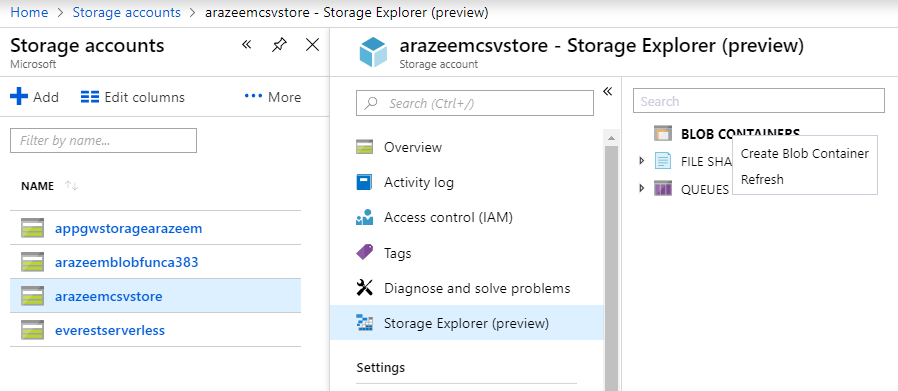


* Under Outputs click **+ New Output**
* Click **Azure Blob Storage** then **Select**
* Set the following Azure Blob Storage Output settings:

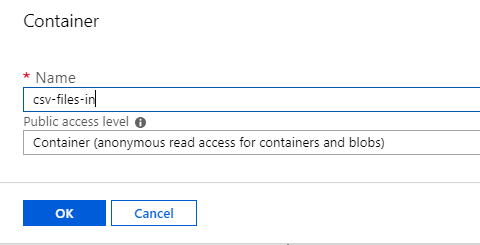
| **Setting** | **Value** | **Description** |
| --- | --- | --- |
| **Blob parameter name** | outputBlob | The variable name used in the function code  to refer to the blob |
| **Path** | csv-files-out/{name}-processed.csv | This is the name of the storage container  where the processed csv files will be placed,  along with a template filename. |
| **Storage account connection** | <The Storage Account You Just Created> | The Storage Account to use. Select the same  account as the input container. |

**Create Blob Container**

1. Go to your storage Account, Select “Storage Explorer (preview)” as show in the following diagram
2. Right Click on Block Containers, and Select “Create Block Container”



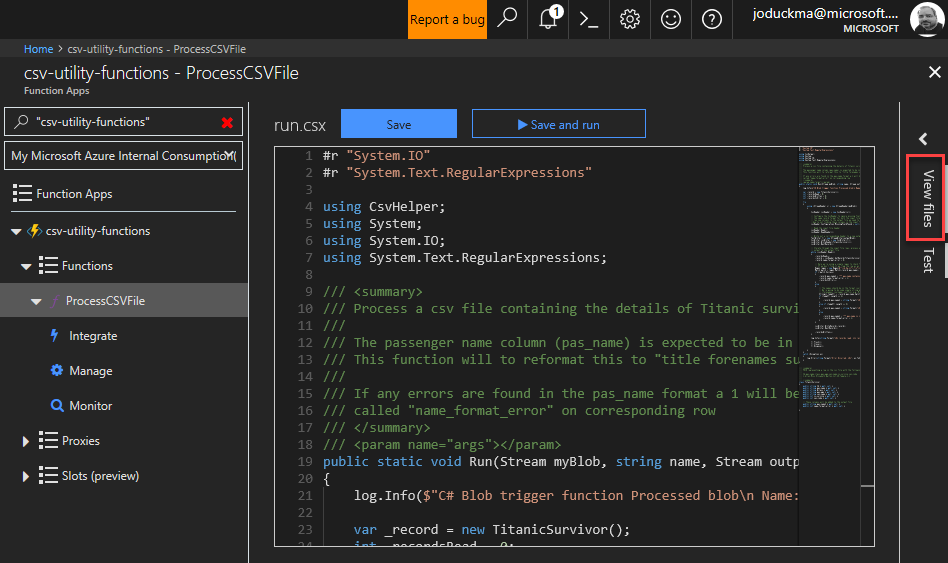
1. Name the container “**csv-files-in**”, and Select “Container (anonymous read access…)” from the drop down as show in the following diagram, and Click “OK”.



1. Now, repeat the above steps to create another container named “**csv-files-out**”.

## Add the function code

* Back in the Azure Portal click on the function **ProcessCSVFile** to open the code editor.
* Replace the contents of the editor window with the code in [this file](https://azurecitadel.github.io/labs/functions/run.csx.txt).
* This code uses a third-party library called [CsvHelper](http://joshclose.github.io/CsvHelper/) to process the csv file. This library is packaged as a [NuGet](http://nuget.org/) package so in order to reference it in our code we need update (or in this case create) a project.json file containing the dependencies:
* Save [this file](https://azurecitadel.github.io/labs/functions/project.json.txt) file to your local machine as **project.json (Note: Please make sure that the file name is NOT project.json.txt, but project.json instead)**
* Click on the **View Files** option on the right of the window



* Click **Upload** and upload the project.json file to Azure. When the upload has completed click on the file. It should look like this…

{

"frameworks": {

"net46":{

"dependencies": {

"CsvHelper": "6.0.0"

}

}

}

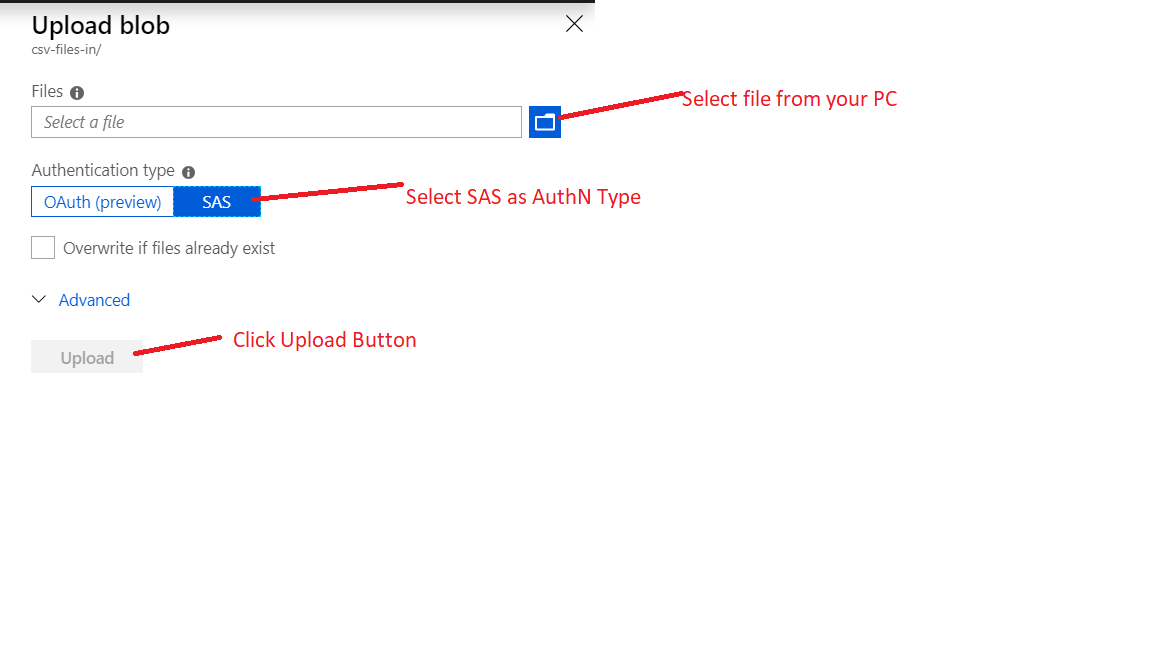
}

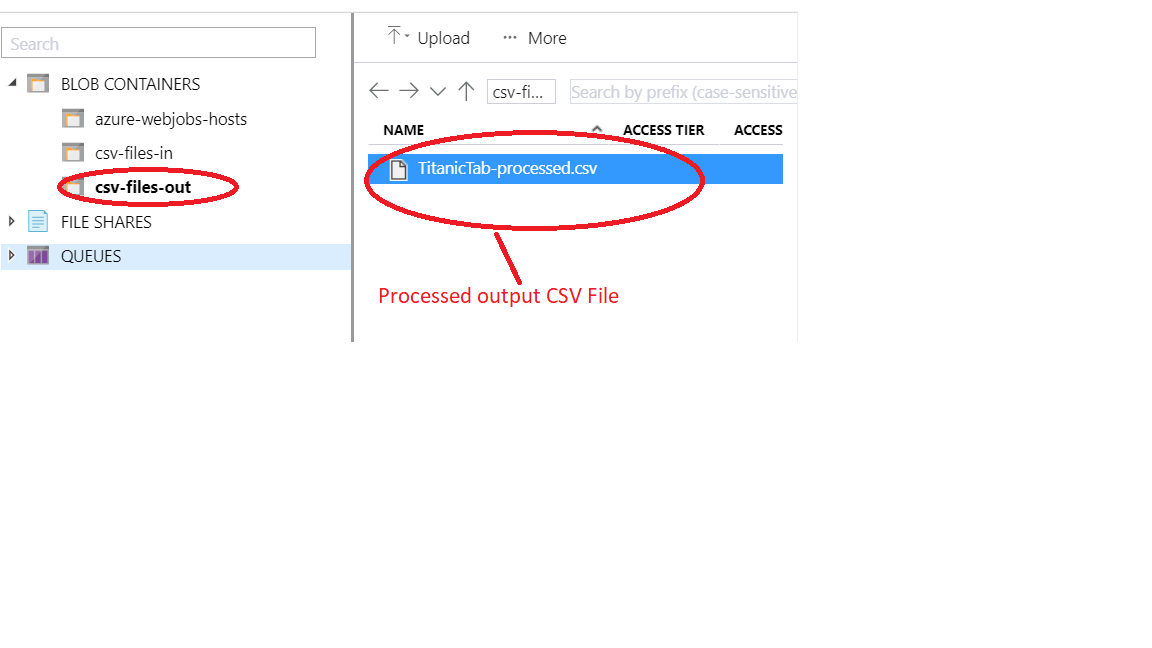
* Click **Save** at the top of the screen. The function will be compiled and the results will be displayed in the log window at the bottom of the window If all is well you should see: Compilation succeeded.

## Testing the function

To test that the function works as expected:

* In the Azure Portal, click on the **ProcessCSVFile** function name and expand the Logs section so that we can see the output when the function runs.
* Download csv data file of [Titanic Survivors](https://azurecitadel.github.io/labs/functions/TitanicTab.csv) to your local PC.
* Go to the Azure Storage Explorer and upload this file to your **csv-files-in** storage container. Select Authentication Type as “SAS”.



* This will trigger your function to execute.
* You should see the function being triggered in the log output, and a file called TitanicTab-processed.csv in your output blob container (you can view and download this file in Storage Explorer).