**Lab 5 –Serverless Image Classification Pipeline**

* Name : Xihai Ren
* Course: CST8921 010
* Professor : Ragini Madaan
* Date : 2025/06/09

CST8921 – Cloud Industry Trends

Lab 5 – Serverless Image Classification Pipeline

## 

## Objective

## Upload an image to Blob Storage → Automatically classify image using Azure Vision API → Save labels to a Table Storage

## Prerequisites:

* Azure Subscription
* Computer Vision Resource (Key + Endpoint)
* Azure Storage Account (Blob + Table)

**Lab Activity Overview:**

## Step 1: Create Blob Storage Container

A screenshot of a computer

AI-generated content may be incorrect.

## Step 2: Enable Event Grid on the Blob Storage

## Step 3: Create an Azure Logic App

* Go to Azure Portal → Create Resource → Logic App (Consumption)
* Choose Blank Logic App

A screenshot of a computer

AI-generated content may be incorrect.

* Add Event Grid Trigger:
* When a blob is created (Event Grid)
* Resource Type: Storage Account
* Event Type: Microsoft.Storage.BlobCreated

A screenshot of a computer

AI-generated content may be incorrect.

## Step 4: Add Action: Call Computer Vision API

## A screenshot of a computer AI-generated content may be incorrect.

* Add HTTP Action
* Set up:

Method: POST

URI:

https://<yourregion>.api.cognitive.microsoft.com/vision/v3.2/analyze?visualFeatures=Tags

Headers:

{

"Content-Type": "application/json",

"Ocp-Apim-Subscription-Key": "<your-vision-api-key>"

}

Body:

{

"url": "@{triggerBody()?['data']['url']}"

}

A screenshot of a computer

AI-generated content may be incorrect.

## Step 5: Add Action: Insert Record to Azure Table Storage

## Add Azure Table Storage action:

## Choose Insert Entity

## Table name: e.g., imagetags

## Partition Key: e.g., "images"

## Row Key: e.g., @{triggerBody()?['data']['url']}

## Add properties like:

## {

## "tag1": "@{body('HTTP')['tags'][0]['name']}",

## "tag2": "@{body('HTTP')['tags'][1]['name']}"

## }

A screenshot of a computer

AI-generated content may be incorrect.

Test the Workflow

1. Upload an image to the Blob Storage container
2. Logic App will auto-trigger
3. Vision API classifies image
4. Tags are stored in Azure Table Storage

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Deploy-AIImagePipeline.ps1

# Variables

$resourceGroup = "rg-image-classifier"

$location = "eastus"

$storageAccount = "imgclassifystorage$((Get-Random -Minimum 1000 -Maximum 9999))"

$containerName = "images"

$tableName = "imagetags"

$computerVisionName = "cvimgclassifier$((Get-Random -Minimum 1000 -Maximum 9999))"

$logicAppName = "logicapp-img-classifier"

$templateFile = "logicapp.json"

# Login if needed

Connect-AzAccount

# Create Resource Group

New-AzResourceGroup -Name $resourceGroup -Location $location

# Create Storage Account (Blob + Table)

New-AzStorageAccount -ResourceGroupName $resourceGroup `

-Name $storageAccount `

-Location $location `

-SkuName Standard\_LRS `

-Kind StorageV2

$ctx = (Get-AzStorageAccount -ResourceGroupName $resourceGroup -Name $storageAccount).Context

# Create Blob container

New-AzStorageContainer -Name $containerName -Context $ctx -Permission Off

# Create Table storage

$cloudTable = (Get-AzStorageTable -Name $tableName -Context $ctx -ErrorAction SilentlyContinue)

if (-not $cloudTable) {

New-AzStorageTable -Name $tableName -Context $ctx

}

# Create Computer Vision Resource

New-AzCognitiveServicesAccount -ResourceGroupName $resourceGroup `

-Name $computerVisionName `

-Type "ComputerVision" `

-SkuName "S1" `

-Location $location `

-Kind "CognitiveServices"

$cvAccount = Get-AzCognitiveServicesAccount -ResourceGroupName $resourceGroup -Name $computerVisionName

$cvKey = (Get-AzCognitiveServicesAccountKey -ResourceGroupName $resourceGroup -Name $computerVisionName).Key1

# Deploy Logic App (from ARM template)

New-AzResourceGroupDeployment -Name DeployLogicApp `

-ResourceGroupName $resourceGroup `

-TemplateFile $templateFile `

-TemplateParameterObject @{

logicAppName = $logicAppName;

storageAccountName = $storageAccount;

containerName = $containerName;

computerVisionEndpoint = $cvAccount.Endpoint;

computerVisionKey = $cvKey

}

Write-Host "Deployment completed. Upload an image to the 'images' container to test the pipeline."

ARM template to deploy logic app

* Triggers on **BlobCreated** events from Azure Blob Storage
* Calls **Azure Computer Vision API** to classify the image
* Stores the top tags in **Azure Table Storage**

{

"$schema": "https://schema.management.azure.com/schemas/2016-06-01/Microsoft.Logic.json",

"contentVersion": "1.0.0.0",

"parameters": {

"logicAppName": { "type": "string" },

"storageAccountName": { "type": "string" },

"containerName": { "type": "string" },

"computerVisionEndpoint": { "type": "string" },

"computerVisionKey": { "type": "string" }

},

"resources": [

{

"type": "Microsoft.Logic/workflows",

"apiVersion": "2019-05-01",

"name": "[parameters('logicAppName')]",

"location": "[resourceGroup().location]",

"properties": {

"definition": {

"$schema": "https://schema.management.azure.com/providers/Microsoft.Logic/schemas/2016-06-01/workflowdefinition.json#",

"contentVersion": "1.0.0.0",

"parameters": {},

"triggers": {

"BlobCreated": {

"type": "Request",

"kind": "Http",

"inputs": {

"schema": {}

}

}

},

"actions": {

"Analyze\_Image": {

"type": "Http",

"inputs": {

"method": "POST",

"uri": "[concat(parameters('computerVisionEndpoint'), '/vision/v3.2/analyze?visualFeatures=Tags')]",

"headers": {

"Content-Type": "application/json",

"Ocp-Apim-Subscription-Key": "[parameters('computerVisionKey')]"

},

"body": {

"url": "@triggerBody()?['data']['url']"

}

},

"runAfter": {}

},

"Insert\_to\_Table": {

"type": "ApiConnection",

"inputs": {

"method": "post",

"path": "/datasets/default/tables/[parameters('containerName')]/items",

"authentication": {

"type": "Raw",

"scheme": "Key",

"parameter": "<Insert Connection Key Here>" // You must manually create connection or bind with managed identity

},

"body": {

"PartitionKey": "images",

"RowKey": "@triggerBody()?['data']['url']",

"Tag1": "@body('Analyze\_Image')?['tags'][0]?['name']",

"Tag2": "@body('Analyze\_Image')?['tags'][1]?['name']"

}

},

"runAfter": {

"Analyze\_Image": [

"Succeeded"

]

}

}

},

"outputs": {}

},

"parameters": {},

"integrationAccount": {},

"state": "Enabled"

}

}

]

}

## Important Notes:

For grading prepare a lab report with your findings and analysis and share that in an Assignments tab in Brightspace.