# Lab Computer Vision

## Set up Storage

We'll be using two different stores in Azure for this project - one for storing the raw images, and the other for storing the results of our Cognitive Service calls. Azure Blob Storage is made for storing large amounts of data in a format that looks similar to a file-system, and it is a great choice for storing data like images. Azure Cosmos DB is our resilient NoSQL PaaS solution and is incredibly useful for storing loosely structured data like we have with our image metadata results. There are other possible choices (Azure Table Storage, SQL Server), but Cosmos DB gives us the flexibility to evolve our schema freely (like adding data for new services), query it easily, and integrate quickly into Azure Search.

### Azure Blob Storage

In the Portal, click the "+ New" button (when you hover over it, it will say Create a resource) and then enter storage in the search box and choose Storage account. Select create.

Once you click it, you'll be presented with some fields to fill out.

* Choose your storage account name (lowercase letters and numbers),
* Set Account kind to Blob storage,
* Set Replication to Locally-Redundant storage (LRS) (this is just to save money),
* Use the same Resource Group as above, and
* Set Location to the region that is closest to you from the following list: East US, West US, Southeast Asia, West Europe
* All other defaults are fine.

### Cosmos DB

In the Portal, click the "+ New" button (when you hover over it, it will say Create a resource) and then enter cosmos db in the search box and choose Azure Cosmos DB and click Create.

Once you click this, you'll have to fill out a few fields as you see fit. Set Location to the region that is closest to you from the following list: East US, West US, Southeast Asia, West Europe.

In our case, select the ID you'd like, subject to the constraints that it needs to be lowercase letters, numbers, or dashes. We will be using the SQL API so we can create a document database that is queryable using SQL syntax, so select SQL as the API. Let's use the same Resource Group as we used for our previous steps, and the same location, select Pin to dashboard to make sure we keep track of it and it's easy to get back to, and hit Create.

## Collect the Keys

Over the course of this lab, we will collect Cognitive Services keys and storage keys. You should save all of them in a text file so you can easily access them in future labs.

Cognitive Services Keys

Computer Vision API:

Storage Keys

Azure Blob Storage Connection String:

Cosmos DB URI:

Cosmos DB key:

In addition, you will need to add the keys to the settings.json file which is under C:\Hekaton\lab1-computer\_vision\resources\code\Starting-ImageProcessing\TestCLI\settings.json. You will have to replace VisionKeyHere, ConnectionStringHere, CosmosURIHere, and CosmosKeyHere with their corresponding keys that you collect in the next section. Do not change the blob container (images), the database name (images), or the collection name (metadata).

## Image Processing

Once you've set your Cognitive Services API keys, your Azure Blob Storage Connection String, and your Cosmos DB Endpoint URI and Key in your TestCLI's settings.json, you can run the TestCLI.

Run TestCLI, then open Command Prompt and navigate to "C:\AzureML\1-Storage\resources\code\Finished-ImageProcessing\TestCLI\bin\Debug" folder (Hint: use the "cd" command to change directories). Then enter TestCLI.exe. You should get the following result:

> TestCLI.exe

Usage: [options]

Options:

-force Use to force update even if file has already been added.

-settings The settings file (optional, will use embedded resource settings.json if not set)

-process The directory to process

-query The query to run

-? | -h | --help Show help information

By default, it will load your settings from settings.json (it builds it into the .exe), but you can provide your own using the -settings flag. To load images (and their metadata from Cognitive Services) into your cloud storage, you can just tell TestCLI to -process your image directory as follows:

TestCLI.exe -process C:\AzureML\1-Storage\resources\sample\_images

Once it's done processing, you can query against your Cosmos DB directly using TestCLI as follows:

TestCLI.exe -query "select \* from images"