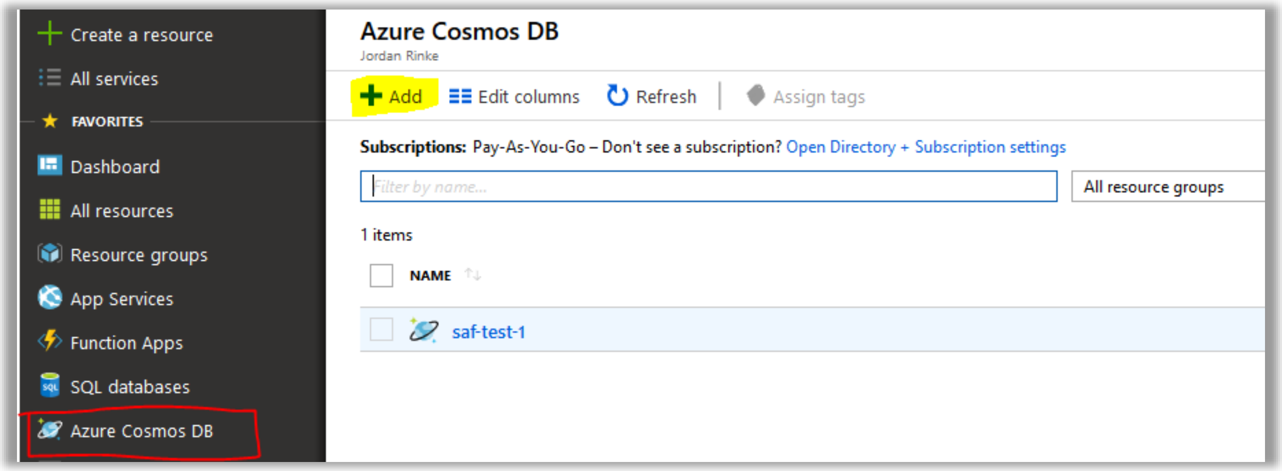
Lab 2: Connecting your Azure Function to Cosmos DB

On the right hand side of the portal select the “Azure Cosmos DB” item and click on “+ Add” to create a new service account for Cosmos:

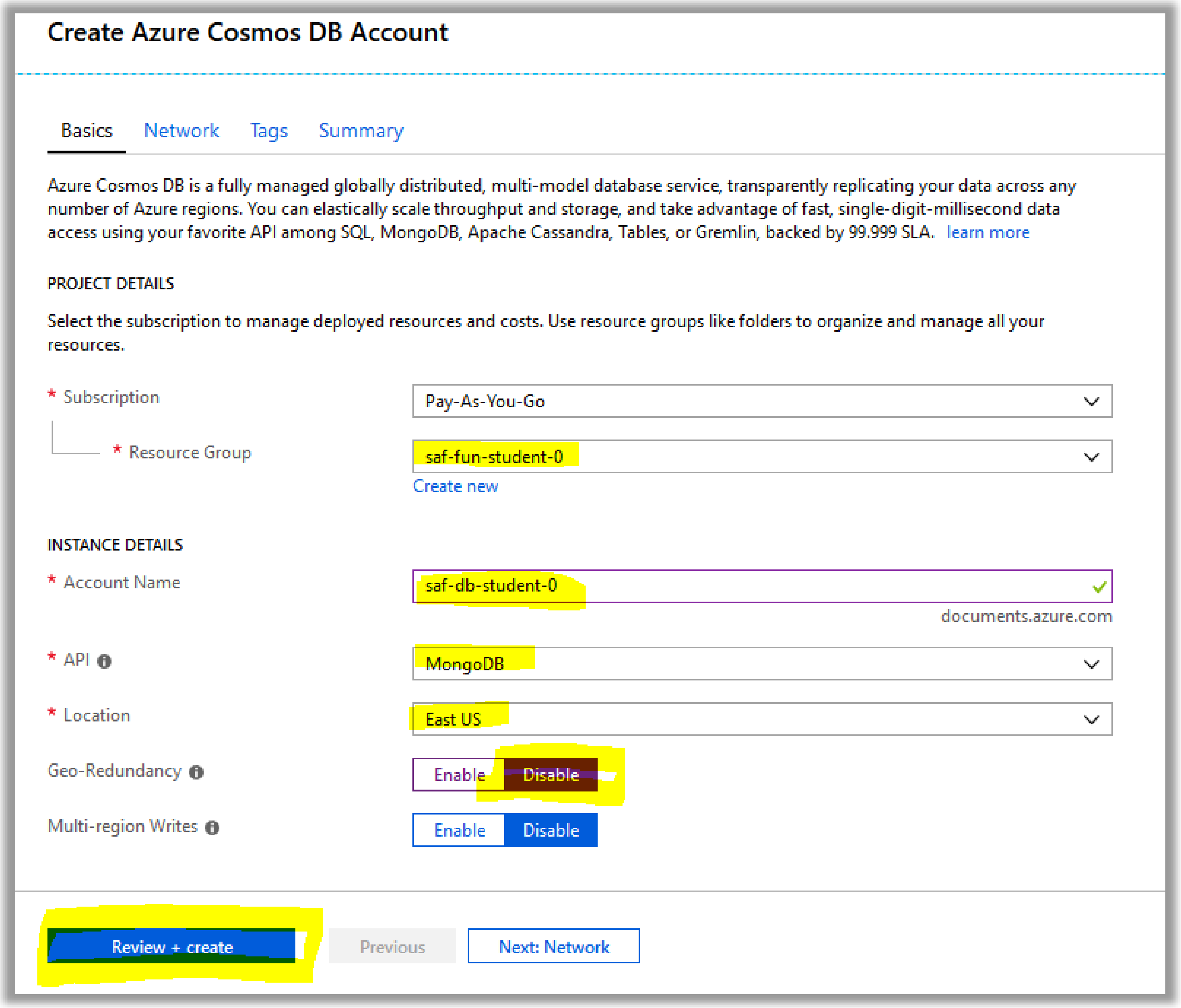


Subscriptions: Pay-As-You-Go – Don't see a subscription? Open Directory + Subscription settings

1 items

saf-test-1

It is generally a good practice to create a new resource group for any application specific items in order to tie them all together logically. Use the resource group that was created when you made your first function. Give your Cosmos account name a meaningful name based on your student number. Select the "MongoDB API" and pick a location that matches where your functions are running. Disable geo-redundancy as we simply don't need it for the lab. Review and create your new account:



Create Azure Cosmos DB Account

Azure Cosmos DB is a fully managed globally distributed, multi-model database service, transparently replicating your data across any number of Azure regions. You can elastically scale throughput and storage, and take advantage of fast, single-digit-millisecond data access using your favorite API among SQL, MongoDB, Apache Cassandra, Tables, or Gremlin, backed by 99.999 SLA. learn more

PROJECT DETAILS

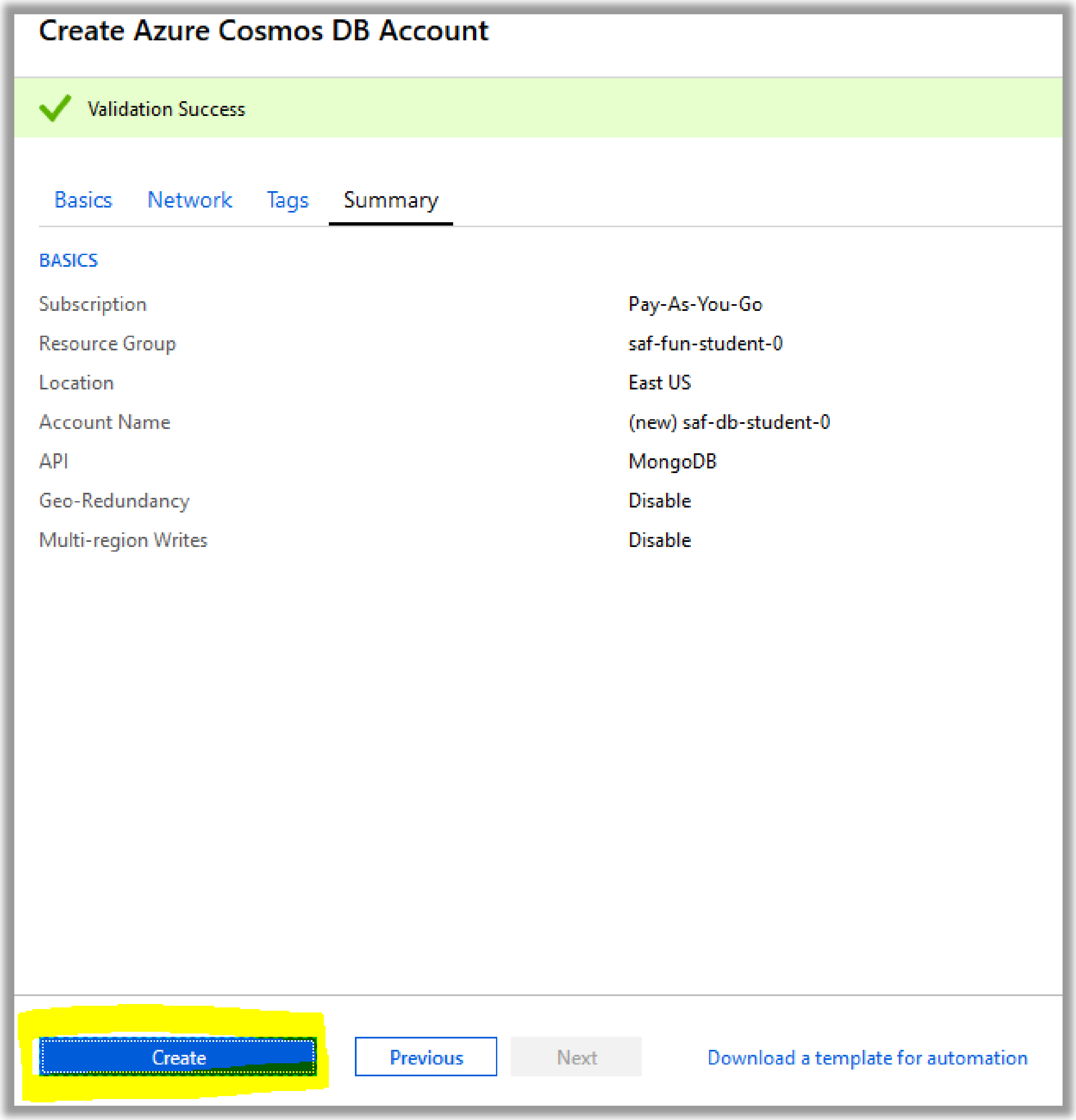
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

* Subscription	Pay-As-You-Go
* Resource Group	saf-fun-student-0
	Create new

INSTANCE DETAILS

* Account Name	saf-db-student-0 ✓
	documents.azure.com
* API	MongoDB
* Location	East US
Geo-Redundancy	Enable [✓ Disable]
Multi-region Writes	Enable [✓ Disable]

When all of the validations have passed, click on "Create":



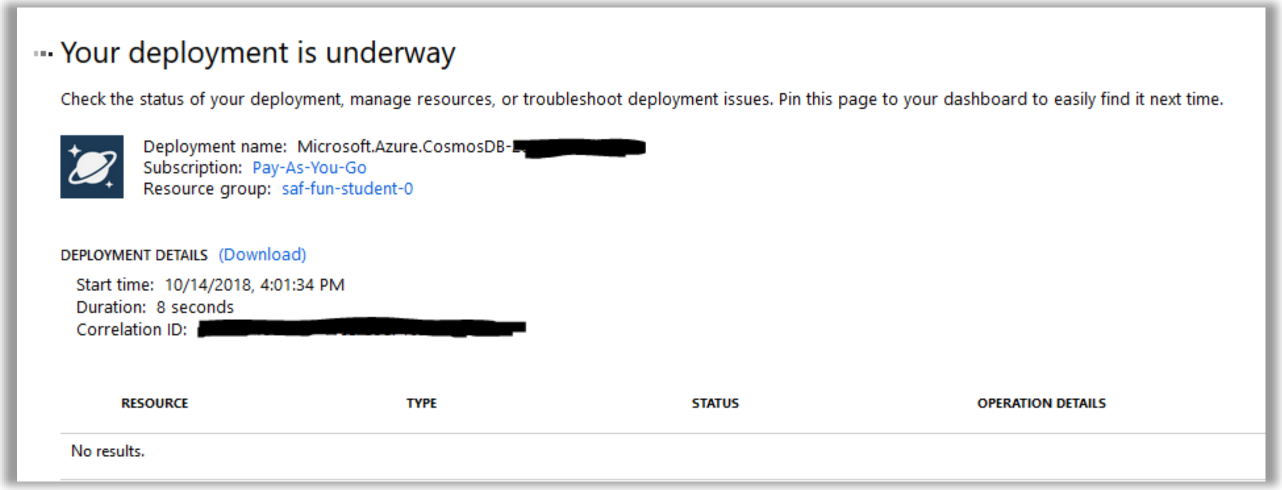
Create Azure Cosmos DB Account

Validation Success

BASICS

Subscription	Pay-As-You-Go
Resource Group	saf-fun-student-0
Location	East US
Account Name	(new) saf-db-student-0
API	MongoDB
Geo-Redundancy	Disable
Multi-region Writes	Disable

Deployment can take a few minutes, once it is done click on Go to resource in order to view the new account and configure the databases we will need for the rest of this lab:



Your deployment is underway

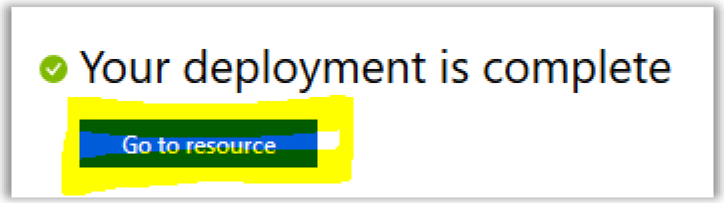
Check the status of your deployment, manage resources, or troubleshoot deployment issues. Pin this page to your dashboard to easily find it next time.

Deployment name: Microsoft.Azure.CosmosDB-[redacted]
Subscription: Pay-As-You-Go
Resource group: saf-fun-student-0

DEPLOYMENT DETAILS (Download)

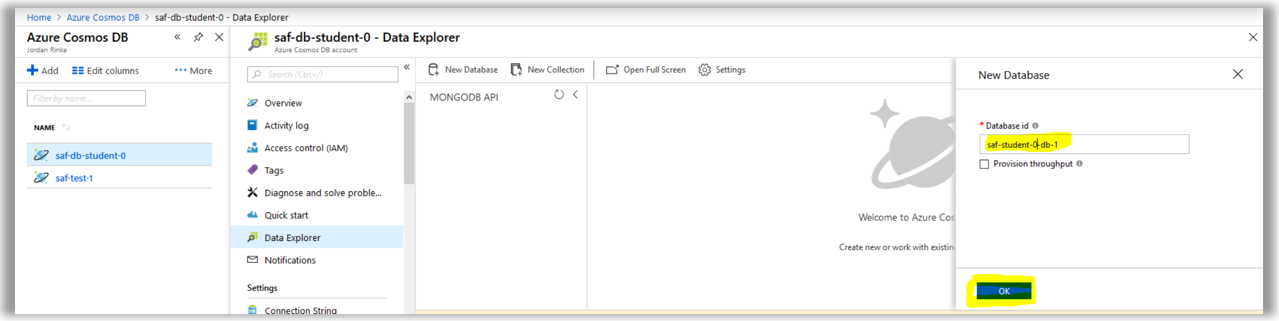
Start time: 10/14/2018, 4:01:34 PM
Duration: 8 seconds
Correlation ID: [redacted]

RESOURCE	TYPE	STATUS	OPERATION DETAILS
No results.			



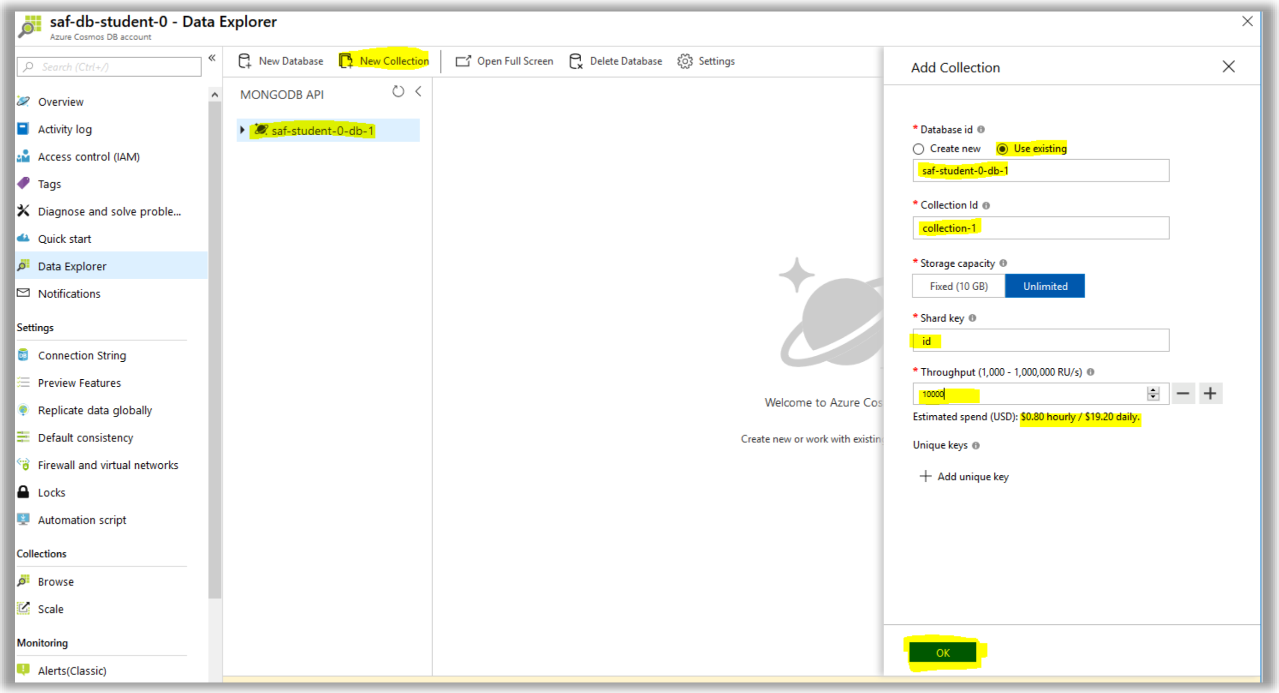
Your deployment is complete

Click on the account you created, select “Data Explorer” and click “New Database” from the top center. Give your database a unique meaningful name based on your student ID. Be sure to remember this, we will need it in a few minutes to connect our function to our database:



* Database id: saf-student-0-db-1
Provision throughput: ☐

Once you have created a database, click on it and then click on “New Collection” select “Use existing” for the database ID and use the name of the database we just created. Give your collection a name, remember this as well, we will need it in a later step. Set a shared key ID. For our lab we will use “id”. Change the throughput allocation to “10,000” and click “OK”:



Add Collection

* Database id
Create new [✓ Use existing]
saf-student-0-db-1

* Collection Id
collection-1

* Storage capacity
Fixed (10 GB) [✓ Unlimited]

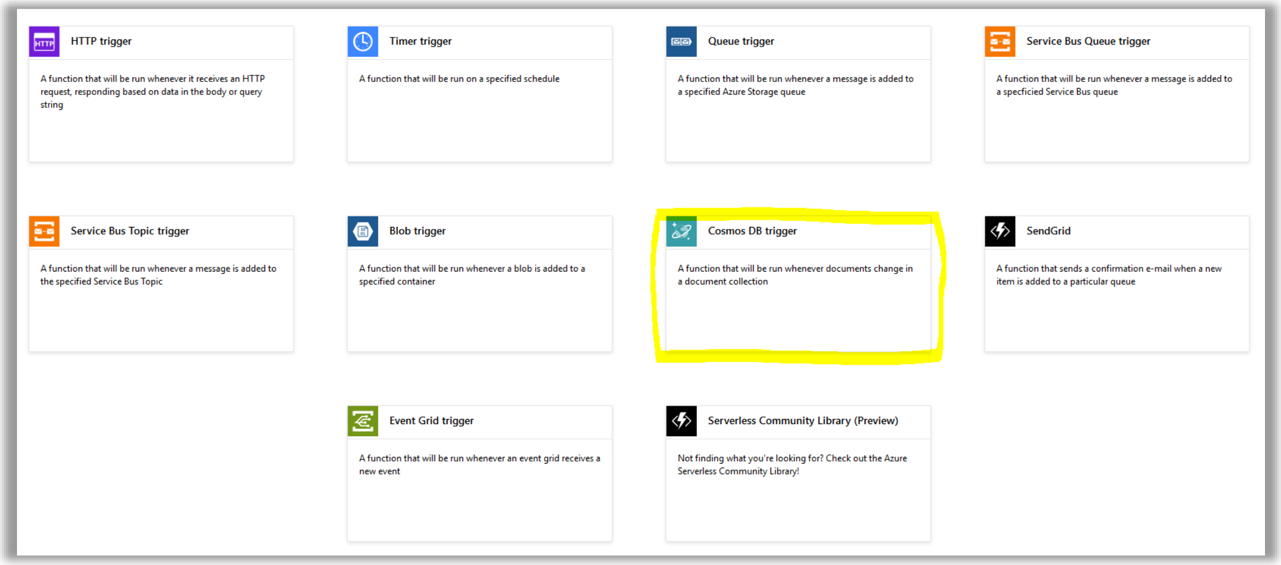
* Shard key
id

* Throughput (1,000 - 1,000,000 RU/s)
10000

Estimated spend (USD): $0.80 hourly / $19.20 daily.

Unique keys

Now that we have a database and a collection, create a new Azure Function. Go back to the functions using the menu on the left, click on your function application name, add a new function and choose the “Cosmos DB Trigger” as your template:



HTTP trigger
A function that will be run whenever it receives an HTTP request, responding based on data in the body or query string

Timer trigger
A function that will be run on a specified schedule

Queue trigger
A function that will be run whenever a message is added to a specified Azure Storage queue

Service Bus Queue trigger
A function that will be run whenever a message is added to a specficied Service Bus queue

Service Bus Topic trigger
A function that will be run whenever a message is added to the specified Service Bus Topic

Blob trigger
A function that will be run whenever a blob is added to a specified container

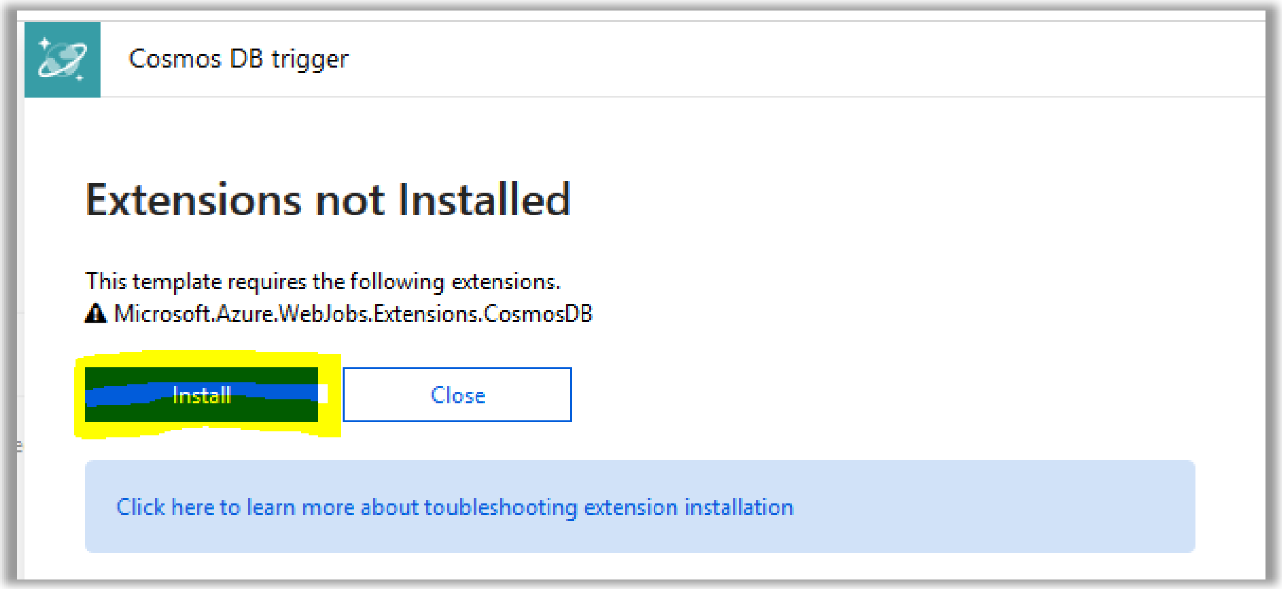
Cosmos DB trigger
A function that will be run whenever documents change in a document collection

SendGrid
A function that sends a confirmation e-mail when a new item is added to a particular queue

Event Grid trigger
A function that will be run whenever an event grid receives a new event

Serverless Community Library (Preview)
Not finding what you're looking for? Check out the Azure Serverless Community Library!

You may be required to install the extension, do so by clicking Install. The installation will take 1-2 minutes:

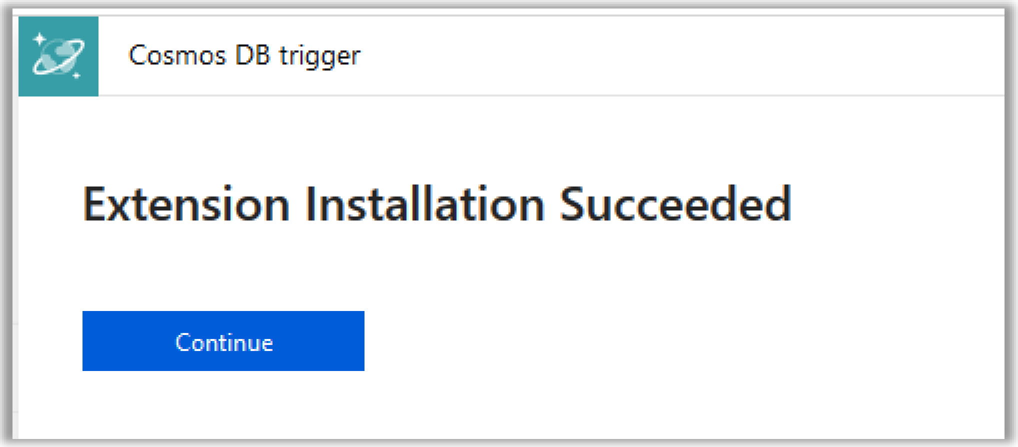

Cosmos DB trigger

Extensions not Installed

This template requires the following extensions.
⚠ Microsoft.Azure.WebJobs.Extensions.CosmosDB

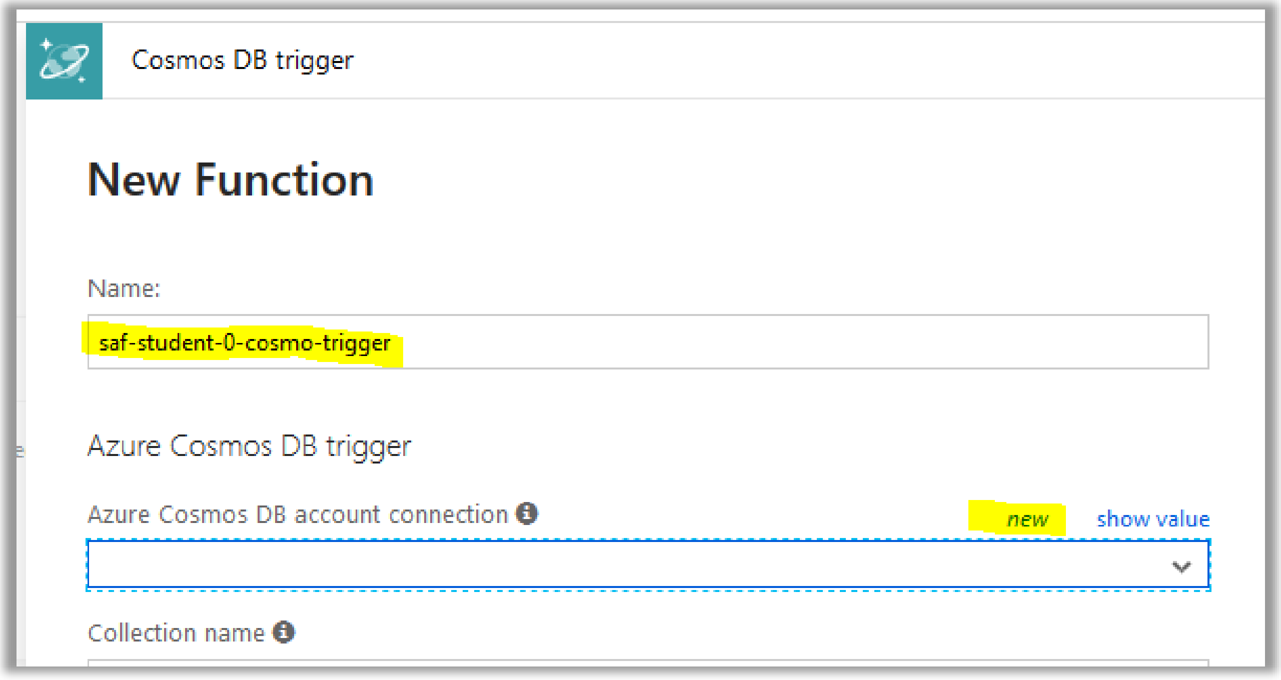
Install
Close
Click here to learn more about toubleshooting extension installation

Once the install has completed, continue:


Cosmos DB trigger

Extension Installation Succeeded

Give your new function a meaningful name based on your student number. Under the DB Account connection select “New”:



Cosmos DB trigger

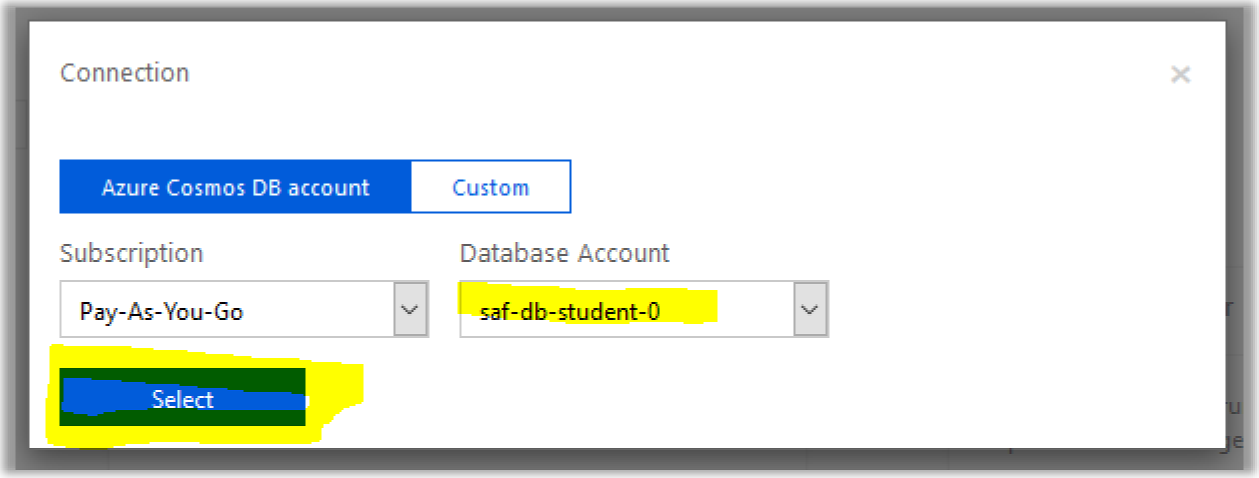
New Function

Name: saf-student-0-cosmo-trigger

Azure Cosmos DB trigger

Azure Cosmos DB account connection

Pick your database account you created earlier from the drop down list and click on “Select”:

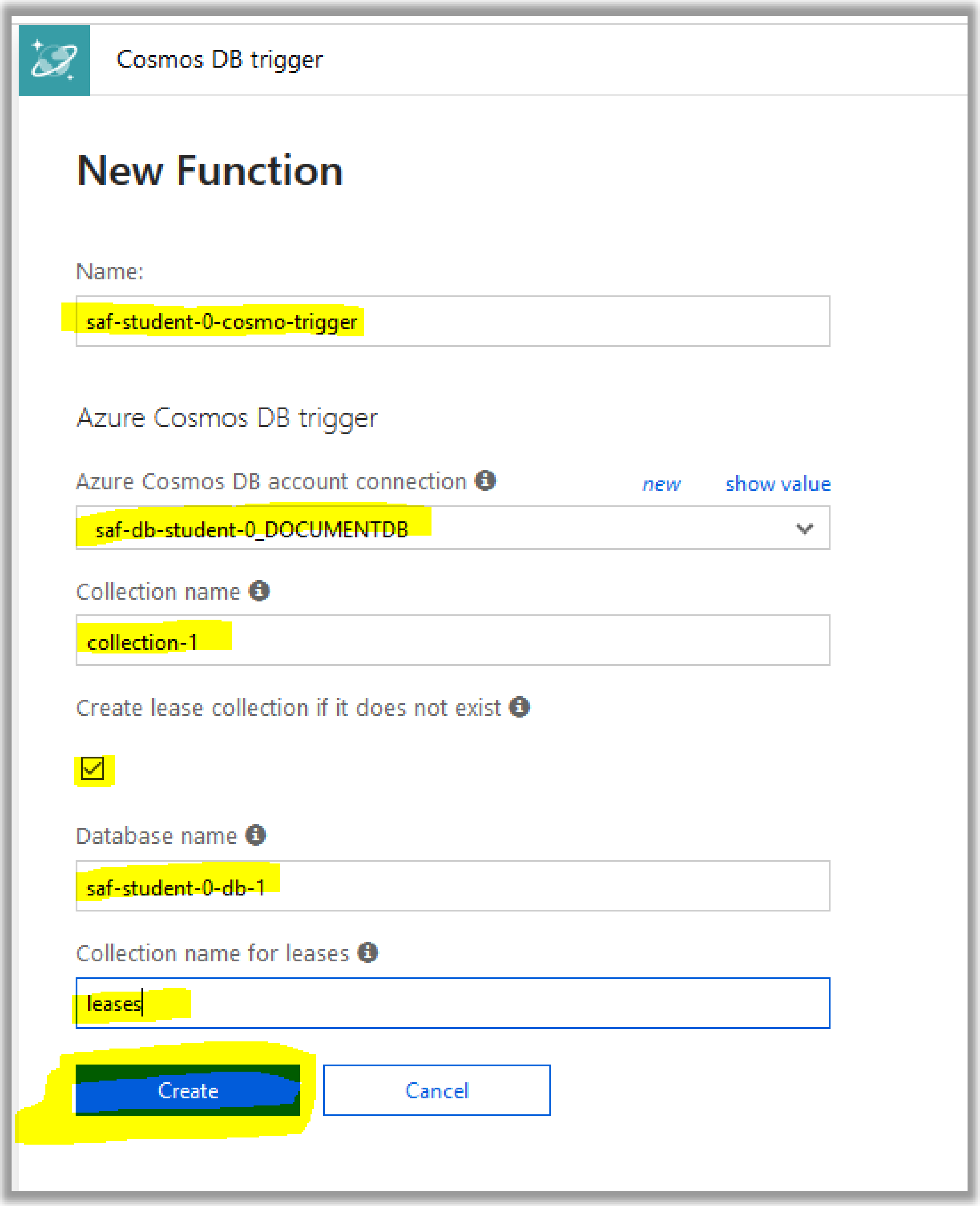


Connection

Subscription: Pay-As-You-Go

Database Account: saf-db-student-0

Finish creating your function by specifying the collection name we created and remembered from earlier. Select the create lease collection option as we will need the function to do this for us. Also specify the database name that we created and made a note to remember from earlier. The collection name for leases should be provided automatically. Click on “Create”:


Cosmos DB trigger

New Function

Name: saf-student-0-cosmo-trigger

Azure Cosmos DB trigger

Azure Cosmos DB account connection: saf-db-student-0_DOCUMENTDB

Collection name: collection-1

Create lease collection if it does not exist: ✓

Database name: saf-student-0-db-1

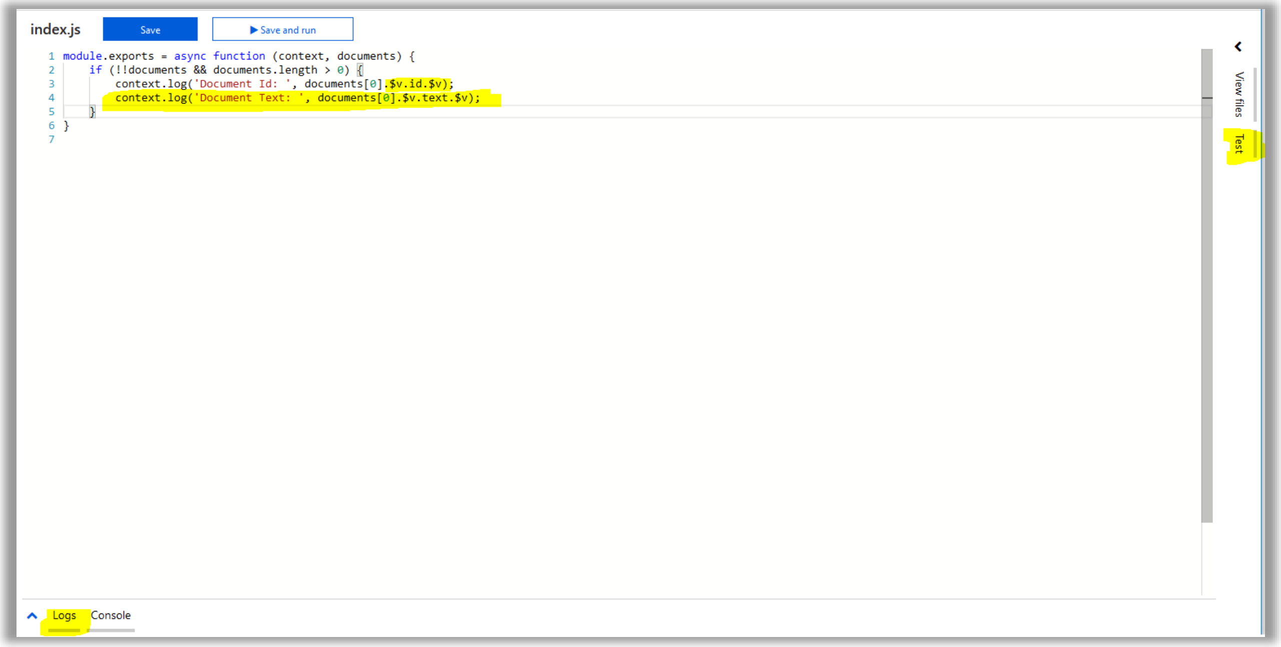
Collection name for leases: leases

Click on the new function to view the code. The code the template provides by default won't work with a Cosmos trigger so let's change it to the following:

```
module.exports = async function (context, documents) {

    context.log(documents);
    if (!!documents && documents.length > 0) {
        context.log('Document Id: ', documents[0].$v.id.$v);
        context.log('Document Text: ', documents[0].$v.text.$v);
    }
}
```

Click "Save" and then click on "Test" on the right hand side and "Logs" on the bottom to expose those panes:

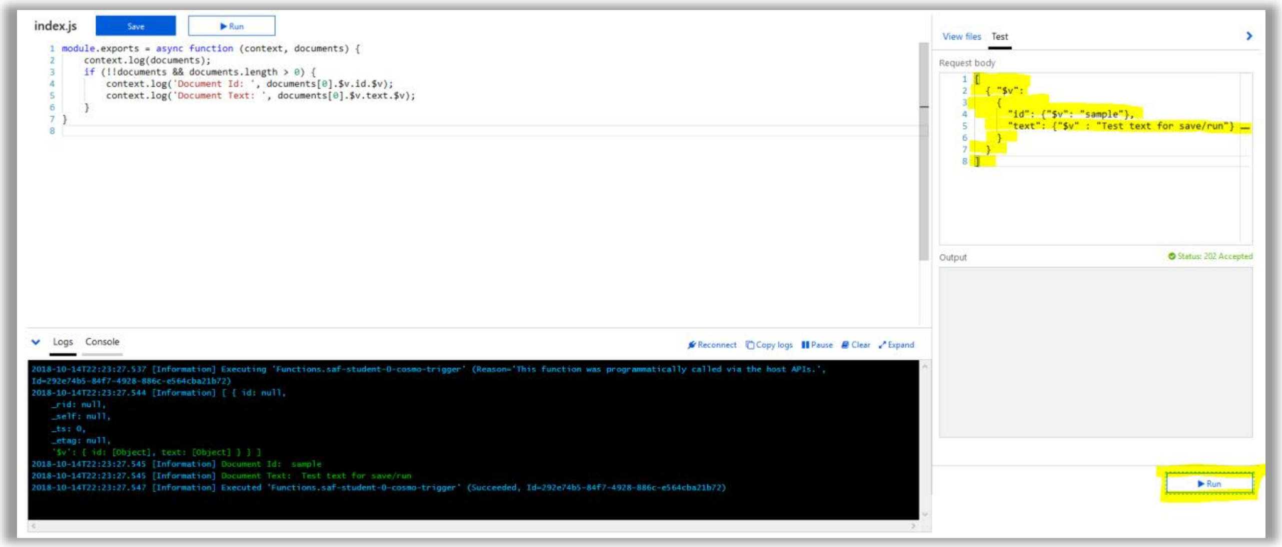


```
module.exports = async function (context, documents) {
    if (!!documents && documents.length > 0) {
        context.log('Document Id: ', documents[0].$v.id.$v);
        context.log('Document Text: ', documents[0].$v.text.$v);
    }
}
```

Change the test data to match what we will get from Cosmos DB:

```
[
        { "$v":
                {
                        "id": {"$v": "sample"},
                        "text": {"$v" : "Test text for save/run"}
                }
        }
]
```

Click on “Test” to see the function run, and the log output in the Log console on the bottom. You should see a message showing the sample Id and test text:

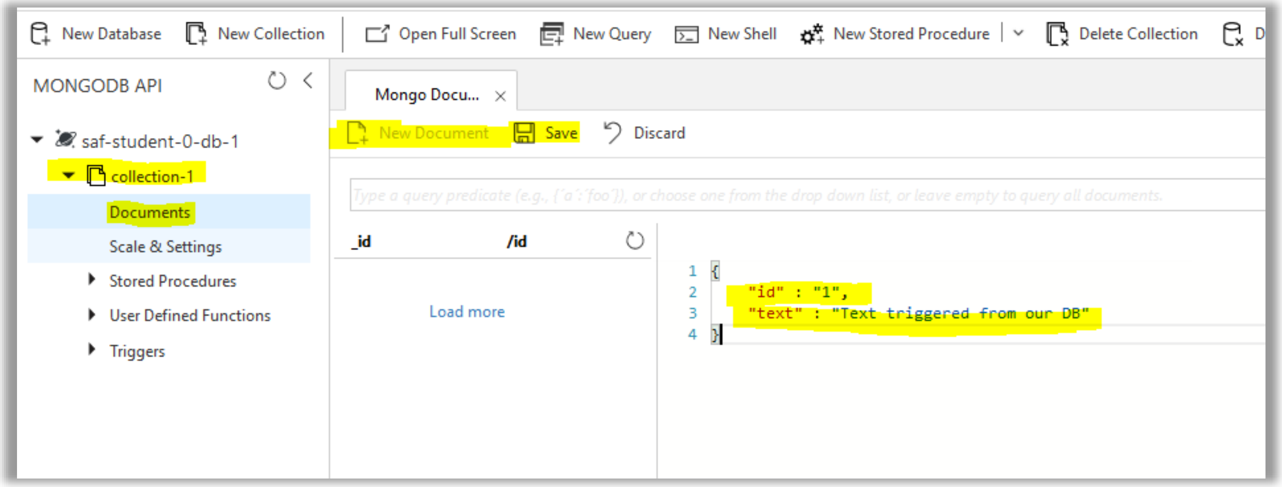


```
module.exports = async function (context, documents) {
    context.log(documents);
    if (!!documents && documents.length > 0) {
        context.log('Document Id: ', documents[0].$v.id.$v);
        context.log('Document Text: ', documents[0].$v.text.$v);
    }
}
```

```
[
  { "$v":
    {
      "id": {"$v": "sample"},
      "text": {"$v" : "Test text for save/run"}
    }
  }
]
```

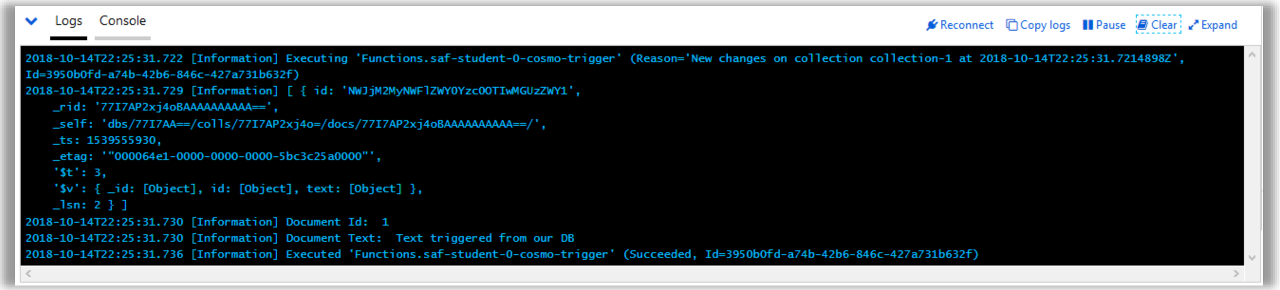
```
2018-10-14T22:23:27.537 [Information] Executing 'Functions.saf-student-0-cosmo-trigger' (Reason='This function was programmatically called via the host APIs.', Id=292e74b5-84f7-4928-886c-e564cba21b72)
2018-10-14T22:23:27.544 [Information] [ { id: null,
    _rid: null,
    _self: null,
    _ts: 0,
    _etag: null,
    '$v': { id: [Object], text: [Object] } } ]
2018-10-14T22:23:27.545 [Information] Document Id:  sample
2018-10-14T22:23:27.545 [Information] Document Text:  Test text for save/run
2018-10-14T22:23:27.547 [Information] Executed 'Functions.saf-student-0-cosmo-trigger' (Succeeded, Id=292e74b5-84f7-4928-886c-e564cba21b72)
```

Now that we have tested the function, let's create a new document. Go back to your database and select the collection. Click on "New Document". Replace the "id" with something unique and add a "text" field with unique text that you will be able to identify in the log output. You will want to open a new tab or window to view the functions log output before you click on "Save". Once you have the log view open click "Save" on the new document:



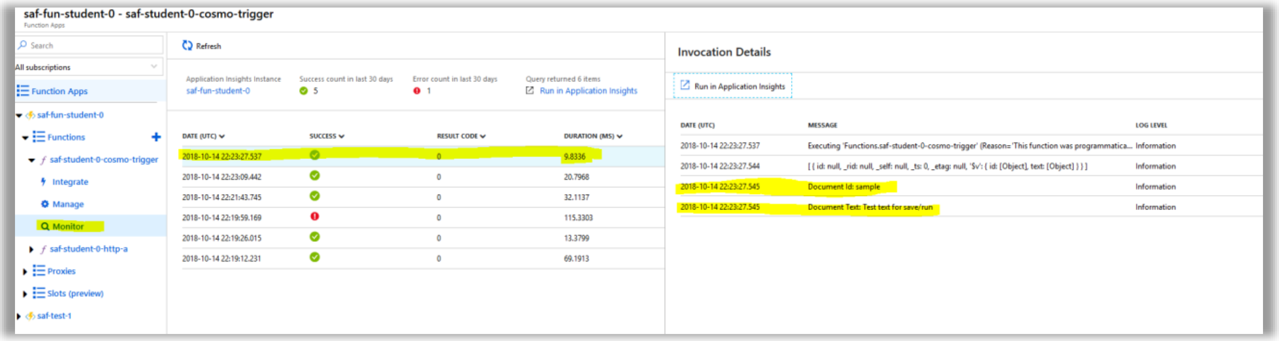
```
{
  "id" : "1",
  "text" : "Text triggered from our DB"
}
```

You should see output in the Log that indicates the event was fired from Cosmos and the relevant data that was passed and displayed in the log:



```
2018-10-14T22:25:31.722 [Information] Executing 'Functions.saf-student-0-cosmo-trigger' (Reason='New changes on collection collection-1 at 2018-10-14T22:25:31.7214898Z',
Id=3950b0fd-a74b-42b6-846c-427a731b632f)
2018-10-14T22:25:31.729 [Information] [ { id: 'NWJjM2MyNWFlZWY0YzcOOTIwMGUzZWY1',
    _rid: '77I7AP2xj4oBAAAAAAAAAA==',
    _self: 'dbs/77I7AA==/colls/77I7AP2xj4o=/docs/77I7AP2xj4oBAAAAAAAAAA==/',
    _ts: 1539555930,
    _etag: '"000064e1-0000-0000-0000-5bc3c25a0000"',
    '$t': 3,
    '$v': { _id: [Object], id: [Object], text: [Object] },
    _lsn: 2 } ]
2018-10-14T22:25:31.730 [Information] Document Id:  1
2018-10-14T22:25:31.730 [Information] Document Text:  Text triggered from our DB
2018-10-14T22:25:31.736 [Information] Executed 'Functions.saf-student-0-cosmo-trigger' (Succeeded, Id=3950b0fd-a74b-42b6-846c-427a731b632f)
```

Clicking on “Monitor” under the function will show you execution details similar to CloudWatch on AWS. The monitoring details here can be delayed by up to 5 minutes so refresh until you see data. Click on an execution to see specific details about it including the output:



Application Insights Instance	Success count in last 30 days	Error count in last 30 days	Query returned 6 items
saf-fun-student-0	✓ 5	● 1	Run in Application Insights

DATE (UTC)	SUCCESS	RESULT CODE	DURATION (MS)
2018-10-14 22:23:27.537	✓	0	9.8336
2018-10-14 22:23:09.442	✓	0	20.7968
2018-10-14 22:21:43.745	✓	0	32.1137
2018-10-14 22:19:59.169	●	0	115.3303
2018-10-14 22:19:26.015	✓	0	13.3799
2018-10-14 22:19:12.231	✓	0	69.1913

Invocation Details

DATE (UTC)	MESSAGE	LOG LEVEL
2018-10-14 22:23:27.537	Executing 'Functions.saf-student-0-cosmo-trigger' (Reason='This function was programmatica...	Information
2018-10-14 22:23:27.544	[{ id: null, _rid: null, _self: null, _ts: 0, _etag: null, '$v': { id: [Object], text: [Object] } }]	Information
2018-10-14 22:23:27.545	Document Id: sample	Information
2018-10-14 22:23:27.545	Document Text: Test text for save/run	Information