CLOUD COMPUTING SYSTEMS

Lab 2

Nuno Preguiça

(nuno.preguica_at_fct.unl.pt)

GOAL

In the end of this lab you should be able to:

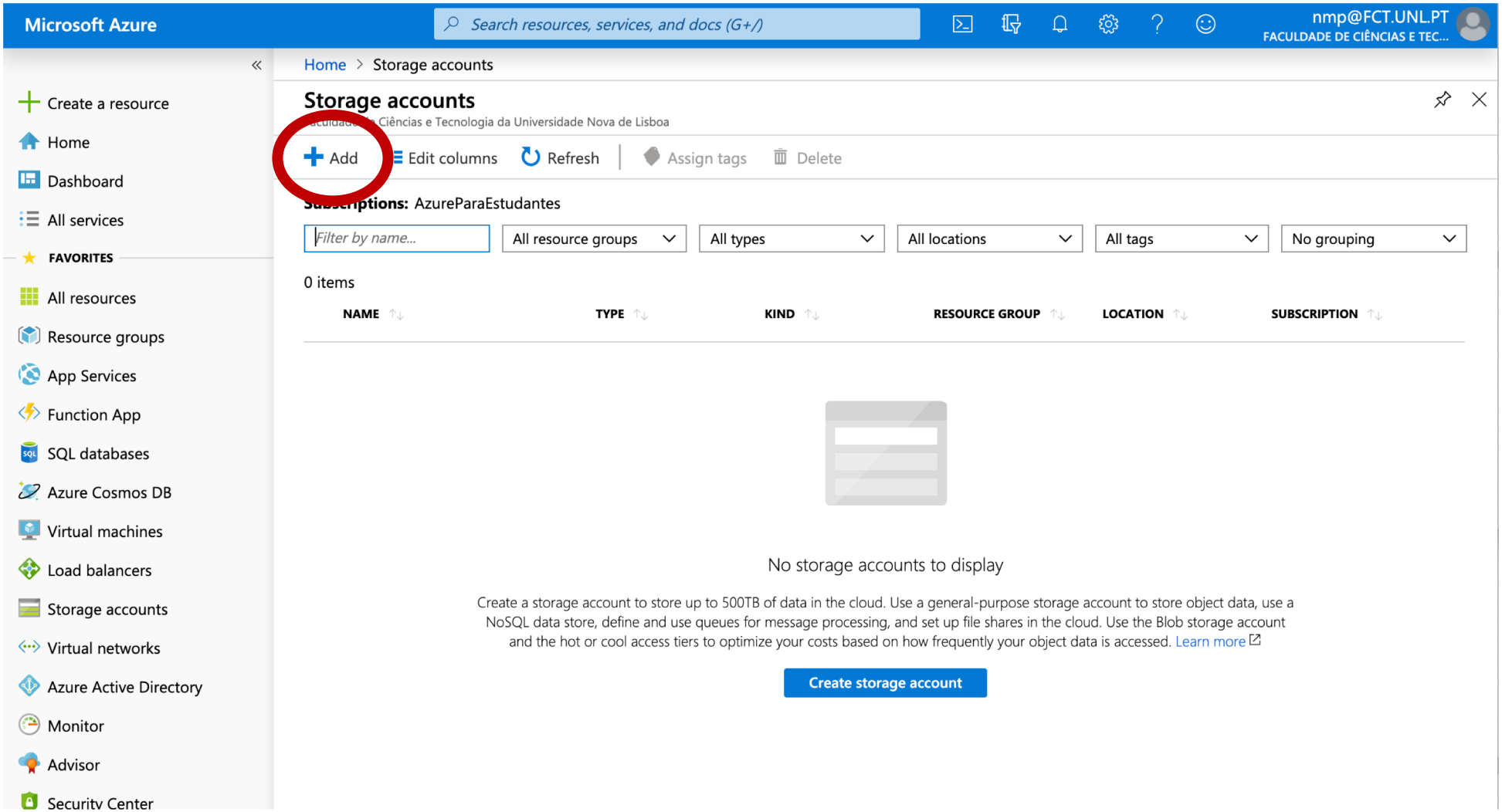
- Create a StorageAccount + Blob Container @ Azure;
- Complete the MediaResource, by storing data at Azure Blob Storage
- Next steps to do for the project…

GOAL

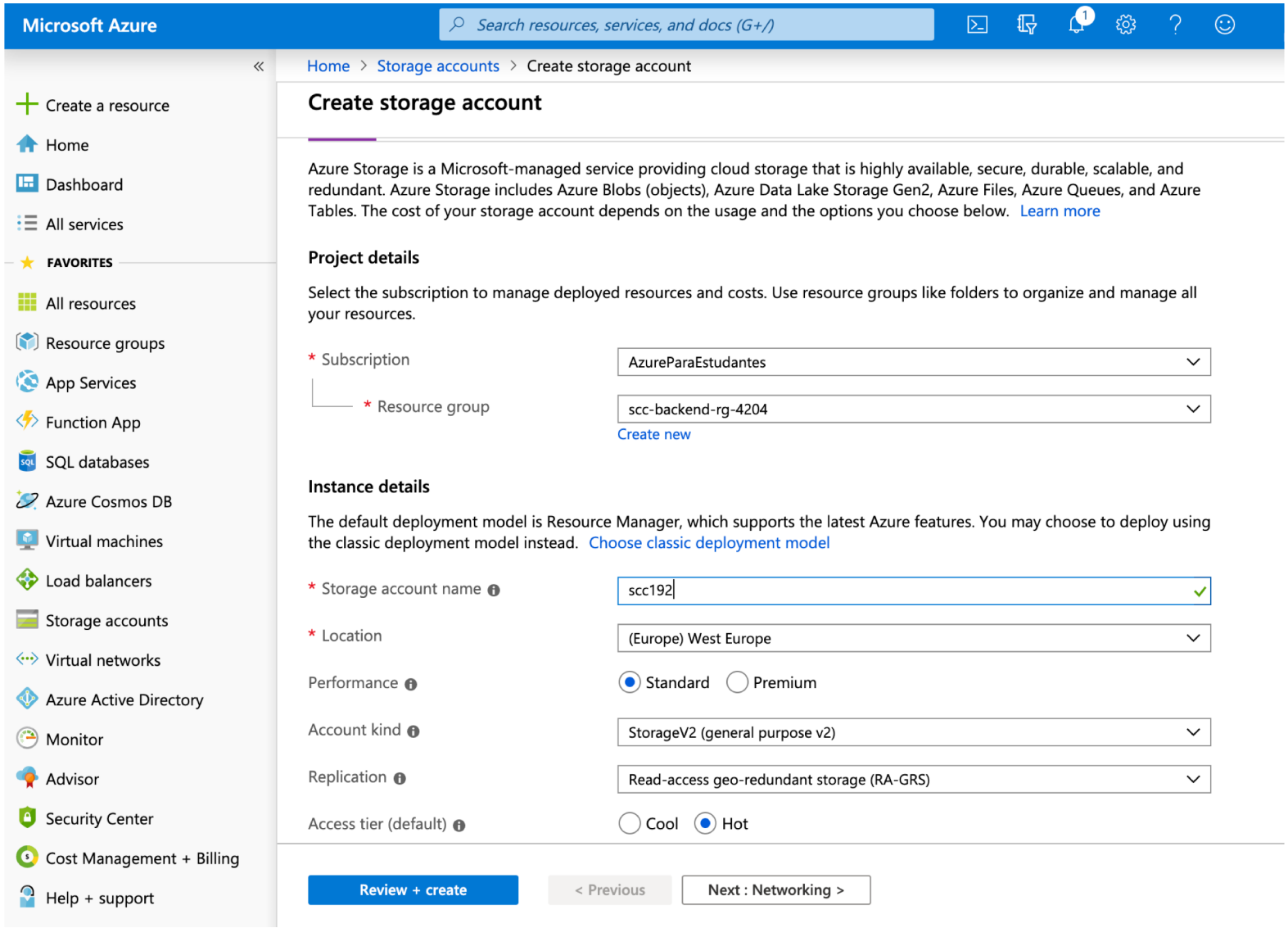
In the end of this lab you should be able to:

- **Create a StorageAccount + Blob Container @ Azure;**
- Complete the MediaResource, by storing data at Azure Blob Storage
- Next steps to do for the project…

CREATE STORAGE ACCOUNT (1)

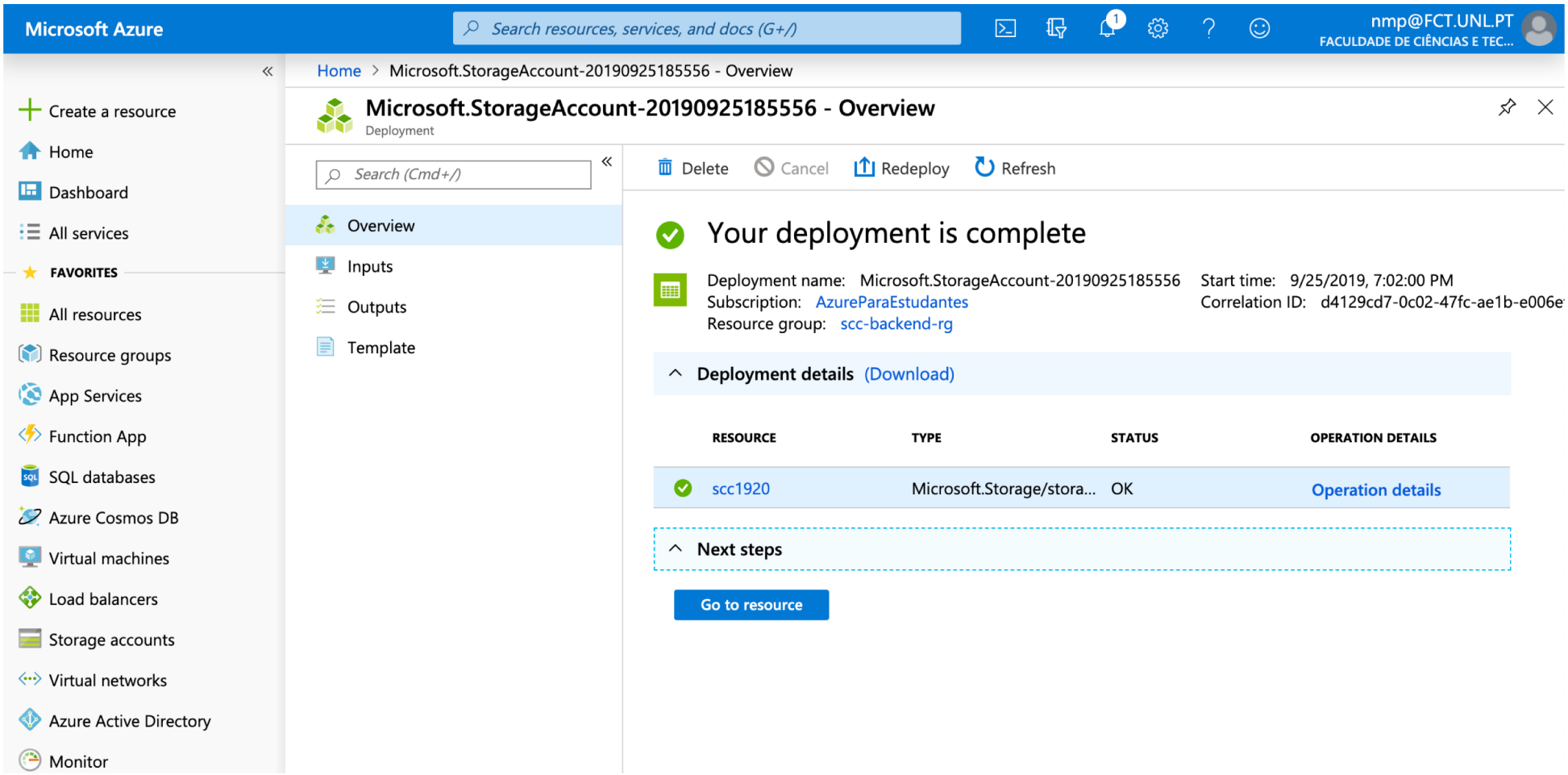


CREATE STORAGE ACCOUNT (2)

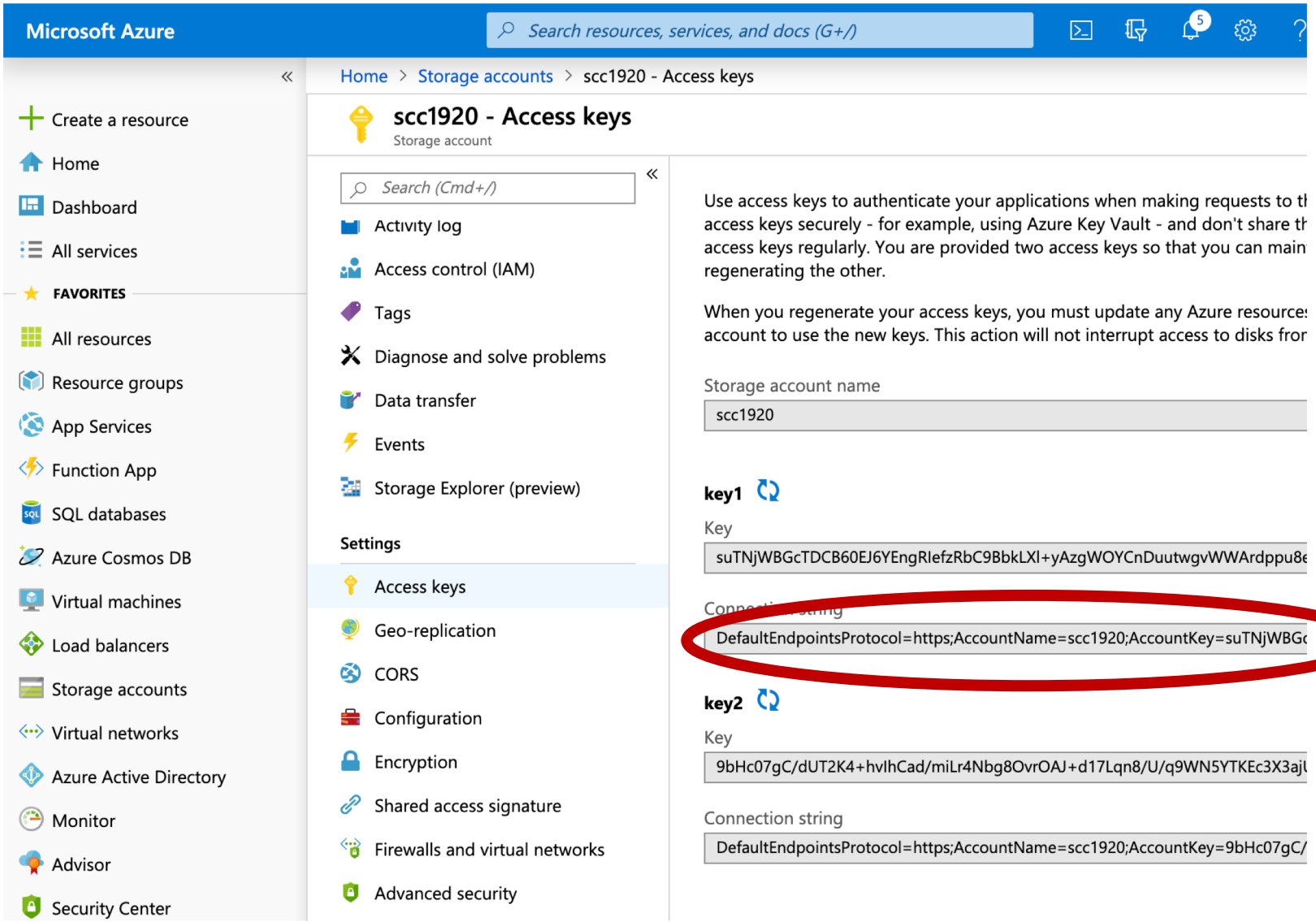


Check lecture 2 for info on this options.

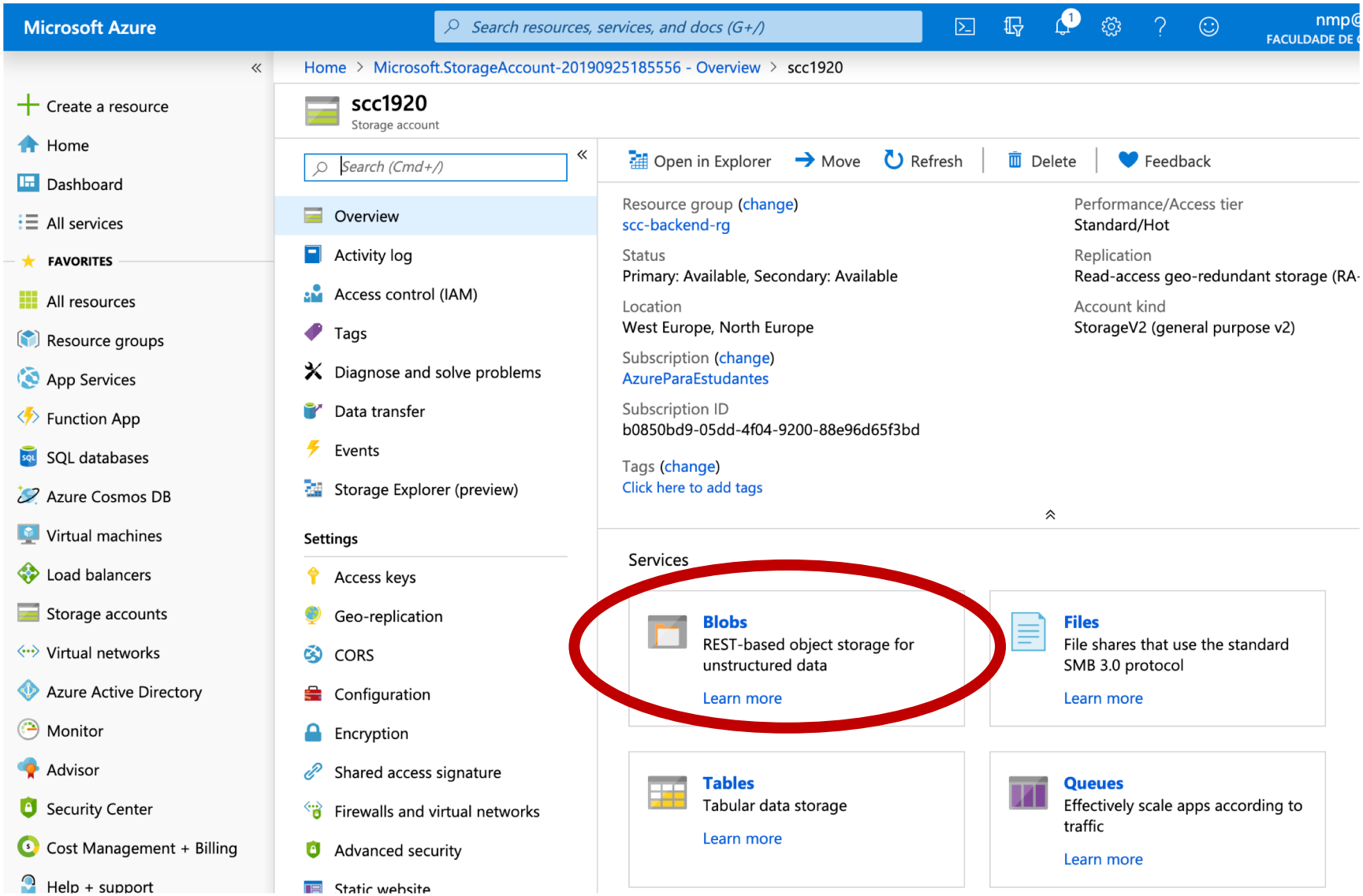
CREATE STORAGE ACCOUNT (3)



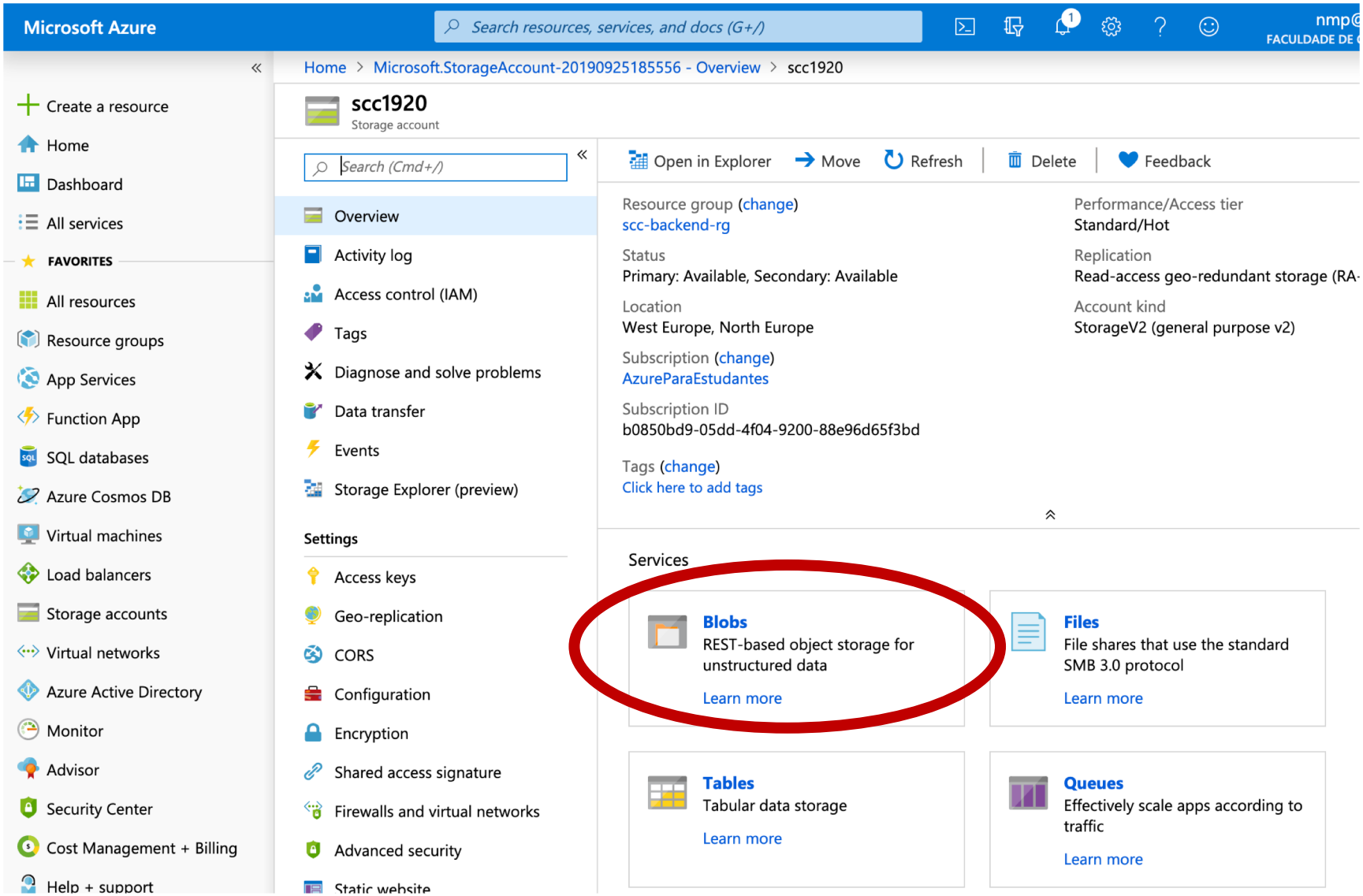
STORAGE ACCOUNT: ACCESS KEYS (FOR CODE)



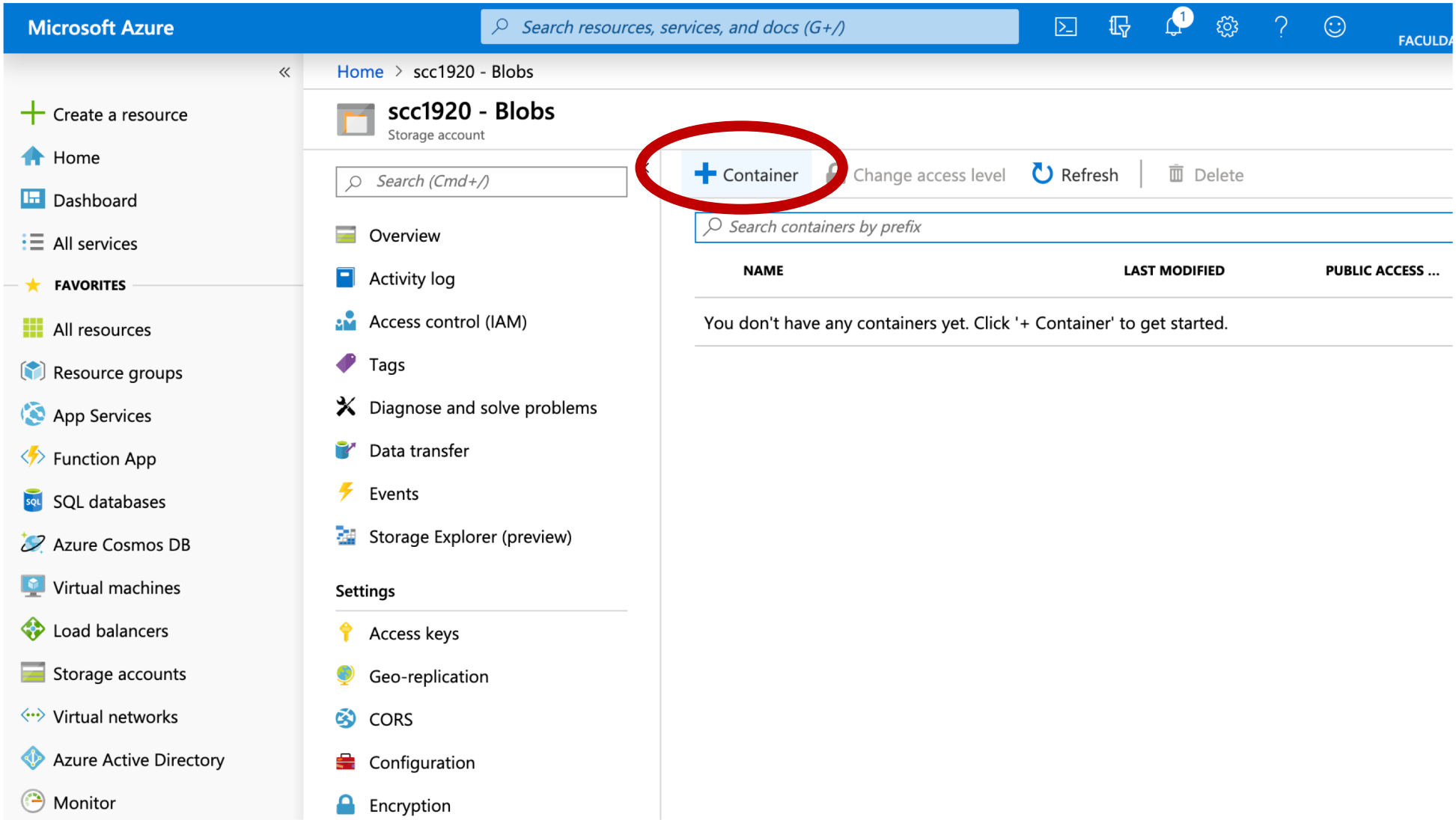
Create Blob Storage Container (1)



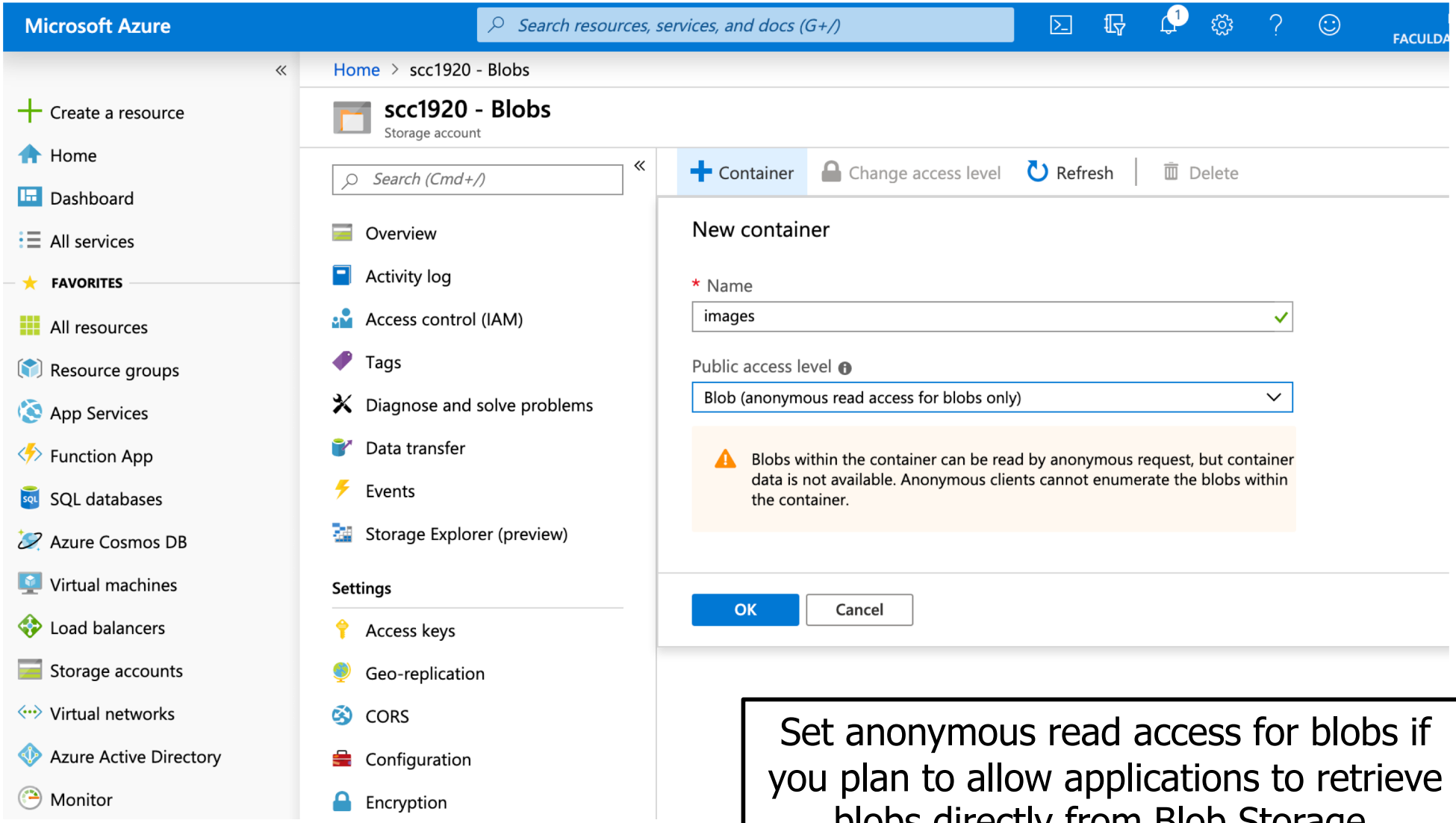
CREATE BLOB STORAGE CONTAINER (2)



CREATE BLOB STORAGE CONTAINER (2)



CREATE BLOB STORAGE CONTAINER (3)

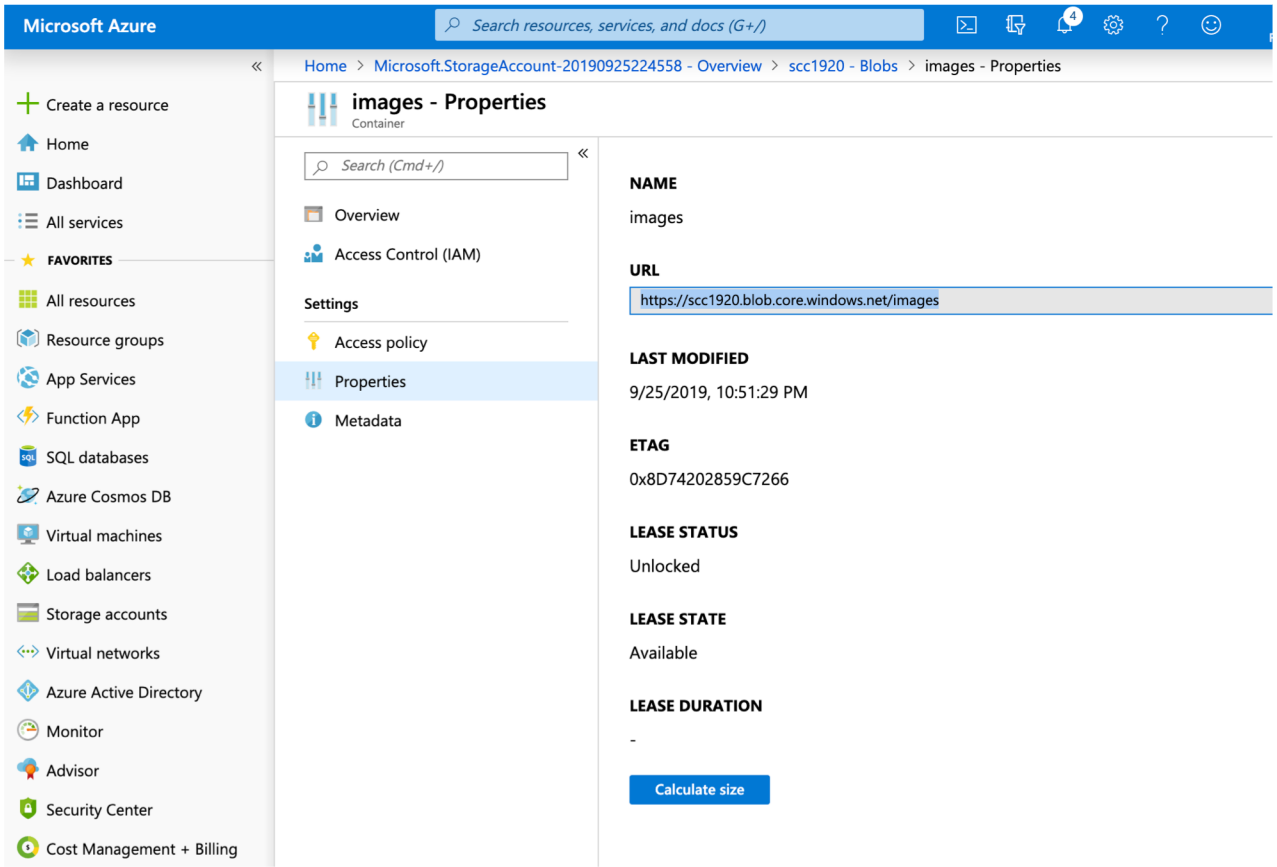


Set anonymous read access for blobs if you plan to allow applications to retrieve blobs directly from Blob Storage.

CONTAINER BASE URL

The base URL allows direct access to the blob by clients.

Try uploading a file (in Azure interface) and downloading it in the browser using this URL.



GOAL

In the end of this lab you should be able to:

- Create a StorageAccount + Blob Container @ Azure;
- **Complete the MediaResource, by storing data at Azure Blob Storage**
- Next steps to do for the project…

ACCESSING AZURE (BLOB) STORAGE

We will use the library provided by Microsoft.

Java Docs available at:

https://javadoc.io/doc/com.microsoft.azure/azure-storage/8.4.0

Example available at:

https://docs.microsoft.com/en-us/azure/storage/blobs/storage-quickstart-blobs-java

Maven dependencies

```
<dependency>
      <groupId>com.microsoft.azure</groupId>
      <artifactId>azure-storage</artifactId>
      <version>8.4.0</version>
</dependency>
```

STEP 1: CREATE CLIENT TO BLOB STORAGE

```
// Get connection string in the storage access keys page
String storageConnectionString = ...


CloudStorageAccount storageAccount =
     CloudStorageAccount.parse(storageConnectionString);
CloudBlobClient blobClient =
     storageAccount.createCloudBlobClient();


CloudBlobContainer container =
     blobClient.getContainerReference("images");
```

STEP 2: UPLOAD BYTE ARRAY

```
// Get reference to blob
CloudBlob blob = container.getBlockBlobReference( name);


// Upload contents from byte array (check documentation
for other alternatives)
blob.uploadFromByteArray(contents, 0, contents.length);
```

STEP 3: DOWNLOAD TO BYTE ARRAY

```
// Get reference to blob
CloudBlob blob = container.getBlobReferenceFromServer(
        name);
ByteArrayOutputStream out = new ByteArrayOutputStream();
blob.download(out);
out.close();

byte[] contents = out.toByteArray();
```

SOME NOTES

- Is it possible to create containers, etc. from the code?

Yes. Check documentation.

GOAL

In the end of this lab you should be able to:

- Create a StorageAccount + Blob Container @ Azure;
- Complete the MediaResource, by storing data at Azure Blob Storage
- **Next steps to do for the project…**

PROJECT

Do the backbone of a system similar to Reddit.

No strict specification, but some features are mandatory.

MULTIMEDIA OBJECT

For each multimedia object, there will be a unique identifier.

POST

For each post, the system must maintain the following information:

- The community;
- The creator;
- Time of creation;
- Text message;
- One link to a multimedia object (optional);
- Reference to the parent post (optional)
  - NOTE: posts without a parent are base posts;
- Number of likes.

USER

For each post, the system must maintain the following information:

- Name.

NOTE: for now, we will ignore authentication.

COMMUNITY

For each community, the system must maintain the following information:

- Name.

PAGES WITH MULTIPLE INFORMATION

The system will have pages that provide multiple data:

- Conversation page: includes the initial post and replies to the post;
- Initial page: includes a selected list of conversations.

TODO

Extend the server to support these new resources.

- Create the necessary endpoint;
  - Base endpoints for multimedia object, post, user and community will be provided.
- Create the objects to represent the resources;
- Start thinking on how to store info on CosmosDB (lecture 3 will introduce the model and lab 3 for code);
- Start thinking on how to provide information for pages with multiple information.