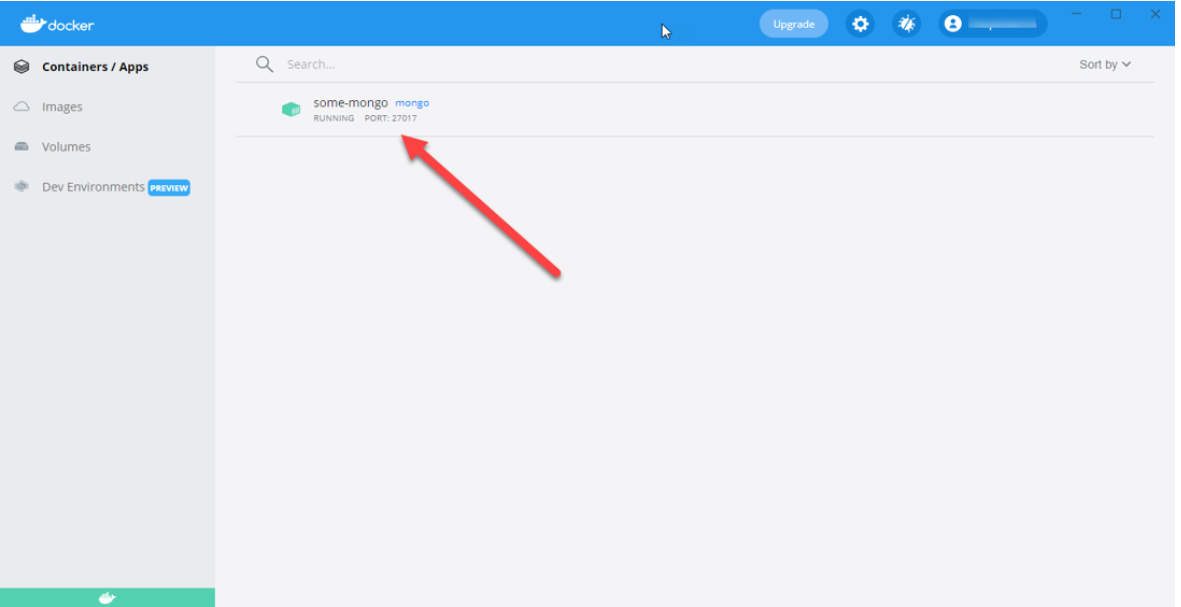
For this activity, you will update data in a MongoDB that is running inside a Docker *container*. To complete this activity, make sure that you have installed the MongoDB libraries for Python by running pip3 install pymongo and that MongoDB is running in your Docker *container* in port 27017. The following command can be used to download the Docker *image* for MongoDB and run it in Docker:

| docker run -p 27017:27017 --name some-mongo -d mongo |
| --- |

If you open your Docker desktop, you should see the some-mongo application running. That’s the MongoDB database you will be interfacing with:

You will perform CRUD (CREATE, READ, UPDATE, and DELETE) *operations* on the MongoDB database.

Before beginning this activity, review the submission instructions below to ensure that you collect the required screenshots as you progress through the activity.

**To complete this activity, follow these steps:**

1. The first step is to create Python code, mongoDBCreate.py. To do this, follow these steps:
   1. Connect to your local MongoDB database.
   2. Connect to the EmployeeDB Remember that for MongoDB you don’t need to create the database in advance. MongoDB will automatically add the database for you if it is not already there.
   3. The employeeCollection is a list of employees in JSON format:

| employeeCollection = [  {"FirstName":"John", "LastName":"Smith", "Age":25},  {"FirstName":"Peter", "LastName":"Smith", "Age":26},  {"FirstName":"Gabriel", "LastName":"Smith", "Age":28},  {"FirstName":"Penny", "LastName":"Lane", "Age":22},  {"FirstName":"Eleanor", "LastName":"Rigby", "Age":23},  {"FirstName":"Helen", "LastName":"Rose", "Age":23}  ] |
| --- |

* 1. Add the following code to verify that the database and *collection* have been created:

| if "employee" in client.list\_database\_names():  print("Employee database created!")  print(result.inserted\_ids) |
| --- |

* 2. Provide a screenshot to show that you successfully created the mongoDBCreate.py file in VS Code. Provide a screenshot of the output to show that the database has been created, “Employee database created!”, and that MongoDB returned the ObjectIds of each of the employees created.

1. The second step is to begin reading the code. For this step, create Python code mongoDBFindOne.py to use the find\_one *method* from the employees *collection*:
   1. Connect to the employee database and look for the employees *collection*.
   2. Invoke the find\_one *method* from the *collection* and pass {“LastName”:”Rigby”} as a parameter, so that the find\_one *method* will return the first record it finds where the last name is “Rigby”:

| employee = collection.find\_one({"LastName":"Rigby"}) |
| --- |

* 2. Print the employee’s record.
  3. Provide a screenshot to show that you successfully created the mongoDBFindOne.py file in VS Code. Provide a screenshot to show the output of the print *statement* with the returned employee with the last name of “Rigby”.

1. For this step, you will continue reading the code. Create Python code mongoDBFindMany.py to use the find *method* from the employees *collection*:
   1. Connect to the employee database and look for the employees *collection*.
   2. Call the find *method* of the *collection* Pass {“LastName”:”Smith”}, which will return a *collection* of all employees with the last name of Smith:

| employeeCursor = collection.find({"LastName":"Smith"}) |
| --- |

* 2. *Loop* through all the employees that were returned and print them.
  3. Provide a screenshot to show that you successfully created the mongoDBFindMany.py file in VS Code. Provide a screenshot to show the output of the code. It should display three employees with the last name “Smith”.

1. For this step, you will begin updating the code. Create Python code mongoDBUpdateOne.py to use the update\_one *method* from the employees *collection*.  
   The update\_one *method* is used to update a record in MongoDB. This *method* has two main arguments you will be using. The first is a *query*, which is the filter for the employees you are interested in. The second is an object defining the new value to be updated. In this part of the activity, you will look for the employee with the last name of “Rose” and will be updating her age to a value of “32”.
   1. Connect to the employee database and look for the employees *collection*.
   2. Create a filter and set it to {“LastName”:”Rose”}:

| filter = {"LastName":"Rose"} |
| --- |

* 2. Create newvalues containing the new values to be updated:

| newvalues = { "$set": { “Age”: 32 } } |
| --- |

* 2. Call the *collection* update\_one *method* and pass the two parameters filter and newvalues:

| collection.update\_one(filter, newvalues) |
| --- |

* 2. Print the employees *collection* and verify that the record has been updated:

| employeeCursor = collection.find()  for employee in employeeCursor:  print(employee) |
| --- |

* 2. Provide a screenshot to show that you successfully created the mongoDBUpdateOne.py file in VS Code. Provide a screenshot to show the output of the code. It should display the employee “Helen Rose” showing Age = 32.

1. For this step, you will continue updating the code. Create Python code mongoDBUpdateMany.py to use the update\_many *method* from the employees *collection*. The update\_many *method* is used to update records in MongoDB. You will also be using two main parameters, filter and newvalues, for this *method*. The update\_many *method* will perform the update on any and all records it finds that satisfy the filter.
   1. Connect to the employee database and look for the employees *collection*.
   2. Create a filter and set it to {“LastName”:”Smith”}:

| filter = {"LastName":"Smith"} |
| --- |

* 2. Create newvalues containing the new values to be updated:

| newvalues = { "$set": { “Department”: “Computer Science” } } |
| --- |

* 2. Call the update\_many *method* and pass the two parameters filter and newvalues:

| collection.update\_many(filter, newvalues) |
| --- |

* 2. Print the employees *collection* and verify that the new *attribute*, Department, has been added to employees with the last name of “Smith”.
  3. Provide a screenshot to show that you successfully created the mongoDBUpdateMany.py file in VS Code. Provide a screenshot to show the output of the code. It should display all employees and the employees with the last name of “Smith” should have a new *attribute* named Department with a value of Computer Science.

1. For this step, you will begin deleting code. Create Python code mongoDBDeleteOne.py to use the delete\_one *method* from the employees *collection*. The delete\_one *method* will take a filter parameter and delete the first record it finds in the *collection* that fits the search criteria.
   1. Connect to the employee database and look for the employees *collection*.
   2. Create a filter and set it to {“LastName”:”Rose”}:

| filter = {"LastName":"Rose"} |
| --- |

* 2. Call the delete\_one *method* and pass the filter to it:

| collection.delete\_one(filter) |
| --- |

* 2. Print the employees *collection* and verify that the employee with the last name of “Rose” has been deleted.
  3. Provide a screenshot to show that you successfully created the mongoDBDeleteOne.py file in VS Code. Provide a screenshot to show that the employee with the last name of “Rose” has been deleted.

1. For this step, you will finish deleting some code. Create Python code mongoDBDeleteMany.py to use the delete\_many *method* from the employees *collection*. The delete\_many *method* will take a filter parameter and will delete all the records it finds in the *collection* that fit the search criteria.
   1. Connect to the employee database and look for the employees *collection*.
   2. Create a filter and set it to {“LastName”:”Smith”}:

| filter = {"LastName":"Smith"} |
| --- |

* 2. Call the delete\_many *method* and pass the filter to it:

| collection.delete\_many(filter) |
| --- |

* 2. Print the employees *collection* and verify that the employees with the last name of “Smith” have been deleted.
  3. Provide a screenshot to show that you successfully created the mongoDBDeleteMany.py file in VS Code. Provide a screenshot to show that the employees with the last name of “Smith” have been deleted.

**Submission Instructions:**

Your submission for this assignment should be a Word document that includes the following screenshots, each labeled for the step that the screenshot represents:

1. 1. Provide a screenshot to show that you successfully created the mongoDBCreate.py file in VS Code.
   2. Provide a screenshot of the output from mongoDBCreate.py to show that the database has been created (“Employee database created!”). This screenshot should also show that MongoDB returned the ObjectIds of each of the employees created.
2. 1. Provide a screenshot to show that you successfully created the mongoDBFindOne.py file in VS Code.
   2. Provide a screenshot to show the output of the print *statement* with the returned employee with last name of “Rigby”.
3. 1. Provide a screenshot to show that you successfully created the mongoDBFindMany.py file in VS Code.
   2. Provide a screenshot to show the output of the code that displays three employees with the last name “Smith”.
4. 1. Provide a screenshot to show that you successfully created the mongoDBUpdateOne.py file in VS Code.
   2. Provide a screenshot to show the output of the code that displays the employee “Helen Rose” showing Age = 32.
5. 1. Provide a screenshot to show that you successfully created the mongoDBUpdateMany.py file in VS Code.
   2. Provide a screenshot to show the output of the code, which should display all employees and the employees with the last name of “Smith” should have a new *attribute* named Department with a value of Computer Science.
6. 1. Provide a screenshot to show that you successfully created the mongoDBDeleteOne.py file in VS Code.
   2. Provide a screenshot to show that the employee with the last name of “Rose” has been deleted.
7. 1. Provide a screenshot to show that you successfully created the mongoDBDeleteMany.py file in VS Code.
   2. Provide a screenshot to show that the employees with the last name of “Smith” have been deleted.