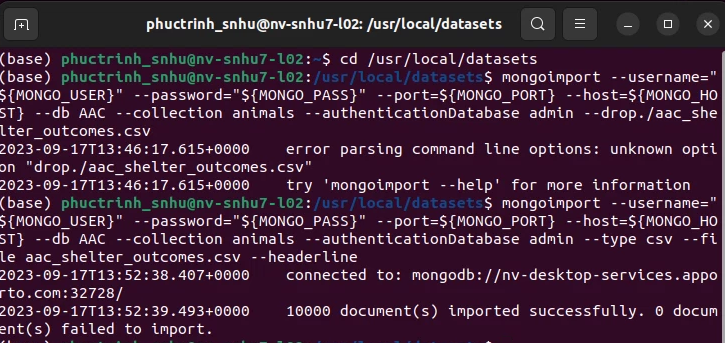
Philip Trinh

SNHU/ CS-340

09/22/2023

4-1 Milestone Create and Read in Python

1. Upload the Austin Animal Center (AAC) Outcomes data set into MongoDB by **importing a CSV file using the appropriate MongoDB import tool**. This file is in the /usr/local/datasets/ directory in Apporto; the filename is “aac\_shelter\_outcomes.csv.” Use the database name “AAC” and collection name “animals.” Complete the import using the **mongoimport** tool, and **take screenshots** of both the import command and its execution.  
   **Note:** If you completed the Module Three milestone, you have already completed this step.

(Previous Screen shot from Module Three milestone)

1. Next, you must develop a Python module in a PY file using object-oriented programming methodology to enable **create** and **read** functionality for the database. Other Python scripts must be able to import your Python code as a module to support code reusability.  
   **Develop** a CRUD class that, when instantiated, provides the following functionality:
   1. **A method that inserts a document into a specified MongoDB database and collection**
      1. Input argument to function will be a set of key/value pairs in the data type acceptable to the MongoDB driver insert API call
      2. Return “True” if successful insert, else “False”
   2. **A method that queries for documents from a specified MongoDB database and collection**
      1. Input arguments to function should be the key/value lookup pair to use with the MongoDB driver find API call
      2. Return result in a list if the command is successful, else an empty list..

**Important**: Be sure to use **find()** instead of **find\_one()** when developing your method. **Hint:** You must work with the MongoDB cursor returned by the find() method.

As you develop your code, be sure to **use industry standard best practices** such as proper naming conventions, exception handling, and in-line comments. Doing so will ensure that your code is easy to read and reusable for future projects.  
  
**Tip:** Use the following sample code to get started. The authentication to MongoDB is in the initialization method for the CRUD class.

**Example Python Code to Insert a Document**

from pymongo import MongoClient

from bson.objectid import ObjectId

class AnimalShelter(object):

""" CRUD operations for Animal collection in MongoDB """

def \_\_init\_\_(self):

# Initializing the MongoClient. This helps to

# access the MongoDB databases and collections.

# This is hard-wired to use the aac database, the

# animals collection, and the aac user.

# Definitions of the connection string variables are

# unique to the individual Apporto environment.

#

# You must edit the connection variables below to reflect

# your own instance of MongoDB!

#

# Connection Variables

#

USER = 'aacuser'

PASS = 'SNHU1234'

HOST = 'nv-desktop-services.apporto.com'

PORT = 31580

DB = 'aac'

COL = 'animals'

#

# Initialize Connection

#

self.client = MongoClient('mongodb://%s:%s@%s:%d' % (USER,PASS,HOST,PORT))

self.database = self.client['%s' % (DB)]

self.collection = self.database['%s' % (COL)]

# Complete this create method to implement the C in CRUD.

def create(self, data):

if data is not None:

self.database.animals.insert\_one(data) # data should be dictionary

else:

raise Exception("Nothing to save, because data parameter is empty")

# Create method to implement the R in CRUD.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

Description automatically generated

1. Finally, **create a Python testing script in Jupyter Notebooks that imports your CRUD Python module to call and test the create and read instances of CRUD functionality**. Be sure to use the username and password for the “aacuser” account for authentication when instantiating the class. This script should be created in a separate Jupyter Notebook IPYNB file and should import and instantiate an object from your CRUD library to affect changes in MongoDB. After creating your script, execute it in Jupyter Notebook and take screenshots of the commands and their execution.

A screen shot of a computer code

Description automatically generatedA screenshot of a computer

Description automatically generated# The Jupyter test code

A screenshot of a computer

Description automatically generated#RESULT EXPLAIN

True = successful created record for first insert 'animal\_id': "test\_id",

Exception Raised: Due to empty parameter nothing is saved.

Incorrect data exception: No insert due to invalid type

Print from Query of first inserted animal record 'name': "Coffee"…’