# CS 340 README

## About the Project/Create and Read Python

This project aims to develop a Python module that enables CRUD (Create and Read only) functionality for connecting to a MongoDB database. The project was designed using object-oriented programming methodology and facilitates the interaction with the "AAC" database and the "animals" collection. The module will later be used to connect the user-interface component of a web application (such as a dashboard) to the database component in Project Two.

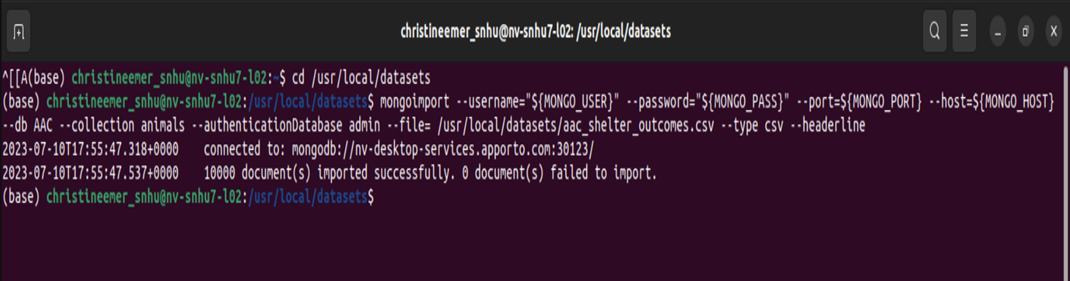
## Motivation

CRUD functionality is essential for interacting with databases, and it forms the backbone of any web application that deals with data storage. The motivation behind creating this Python module is to establish a robust and reusable codebase that provides create and read functionality for the "AAC" database. By adhering to industry standard best practices, such as proper naming conventions, the module will be easily readable and adaptable for future projects.

## Getting Started

To get a local copy up and running of this project, follow these example steps:

1. **Open terminal**
2. **Verify access to the environment by starting up MongoDB and the mongo shell.**
   1. Open the terminal application, which will bring up the Linux shell prompt.
   2. Execute the mongo command to start the mongo shell to bring up the mongo shell prompt, by typing this command:
      1. mongosh
      2. quit – to exit mongosh
3. **Import AAC Data Set CSV File into MongoDB**
   1. Upload the Austin Animal Center (AAC) Outcomes data set into MongoDB by importing a CSV file using the appropriate MongoDB import tool. Use the database name “AAC” and collection name “animals.”
   2. Example code:

****

1. **Open the Mongo Shell and locate the AAC dbs using the command:**
   1. Use AAC
2. **Create a Simple Index Using commands:**
   1. db.animals.createIndex({‘breed’:1})
3. **Create a Compound Index to improve the performance of queries looking for breeds that have an “outcome\_type” of “Transfer”, using the command:**
   1. db.animals.createIndex({‘breed’ : 1, ‘outcome\_type’ : 1})
4. **Create a user name “aacuser” with read/write privileges to that AAC database.**
   1. Use commands:
      1. Use Admin db
      2. Db.createUser({user:’aacuser’, pwd: ‘ ’, roles : [{role : ‘readWrite’, db : AAC}]})
      3. quit
5. **Login to the new user name “aacuser” using the following commands:**
   1. MONGO\_USER = aacuser
   2. MONGO\_PASS = (the password you created)
   3. printenv | grep -i mongo (which shows the users information)
   4. MONGO\_USER = aacuser
   5. MONGO\_HOST = nv-desktop-services.apporto.com
   6. MONGO\_PASS = (the password you created)
   7. MONGO\_PORT = (PORT # you found from b.)
   8. mongosh
6. Open Jupyter Notebook
7. Develop a Python module in a PY file
   1. Open a new text file and rename it CRUD.py
8. Develop a CRUD class to enable **create**
   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 using:
      1. self.database.animals.insert\_one(data)
   2. Return “True” if successful insert, else “False”
9. Enable **read** for the CRUD class
   1. Input arguments to function should be the key/value lookup pair to use with the MongoDB driver find API call using:
      1. searchResult = list(self.database.animals.find(search))
   2. Return result in a list if the command is successful, else an empty list.

**\*\*** I ran into some minimal issues when creating the CRUD class to enable create and read. For example, I over overcomplicated it by creating exceptions and returning true and false. This was making it so that it did not return true or false like I needed it to. Therefore, I removed the exceptions and simply returned true or false and it worked flawlessly.

## Installation

1. pip (we used version 21.2.4 for this project)
   * Go to <https://pypi.org/project/pip/> for download link and instructions on how to install pip.
2. pymongo (we used version 3.12.0 for this project)
   * We recommend using pip to install pymongo on all platforms
   * Enter commands:
     + $ python -m pip install pymongo
     + To get a specific version of pymongo:
     + $ python -m pip install pymongo==3.12.0
   * Go to <https://pymongo.readthedocs.io/en/3.12.3/installation.html> for more information.
3. MongoDB
   * Go to [https://www.mongodb.com/docs/manual/installation/#install-mongodb](https://www.mongodb.com/docs/manual/installation/%23install-mongodb) for instructions on how to install the applicable edition for your platform.
4. Jupyter Notebook
   * We recommend using pip to install Jupyter Notebook
   * Enter commands:
     + Install - python -m pip install jupyter
     + Launch- jupyter notebook
5. mongosh binary
   * Go to <https://www.mongodb.com/docs/mongodb-shell/install/#std-label-mdb-shell-install> to download and more information.
6. Python
   * Go to <https://www.python.org/downloads/> to download and more information.

## Usage

### Code Example

1. create(self,data): function implements Create in Crud, passing a python dictionary of MongoDB data will use insert\_one to create a new db entry.
   1. self.database.animals.insert\_one(data)
2. read(self,search): function implements Read in Crud. If passed a search criterion will search based on that MongoDB query information and the results will return in a list if the command is successful, else an empty list.
   1. *searchResult = list(self.database.animals.find(search))*

### Tests

**To run the test functionality for create and read the following steps are needed:**

1. **from CRUD import AnimalShelter**
2. **Connect to the Database by enter in your password you created** 
   1. animals = AnimalShelter ("aacuser", "PASS")
3. **Create tests code examples:**
   1. # Valid Document Create

created = animals.create({ "age\_upon\_outcome": "6 years", "animal id": "test", "animal\_type": "Dog", "breed": "Golden Retriever", "color": "Golden", "date\_of\_birth": '2020-07-18', "datetime": '020-07-18 01:01:00', "monthyear": '2020-07-18T11:24:00',

"name": 'Diamond', "outcome\_subtype": 'Partner', "outcome\_type": 'Transfer', “sex\_upon\_outcome": 'Spayed Female', "location\_lat": 30.4861149595305, "location\_long": -97.2691161890552, "age\_upon\_outcome\_in\_weeks": 312.857

})

print(created)

* 1. # Invalid Document Create to Test Returning False

test = animals.create(None)

print(test)

1. **Read tests code examples:**
   1. # Invalid Search to Test Showing an Empty List

testQuery2 = animals.read({"name":"peenjut"})

print(testQuery2)

* 1. # Valid search for new document

query = animals.read({"animal id": "test"})

### Screenshots

*A screenshot of a computer

Description automatically generated*

*A screenshot of a computer

Description automatically generated*

## Contact

If you have any questions or need assistance with the Python CRUD module, feel free to reach out:

Your Name: Christine Emerson

Email: christine.emerson@snhu.edu

LinkedIn: <https://www.linkedin.com/in/christine-emerson-b0631b190>