# CS 340 README CRUD Python Module

## About the Project/Project Title

This project shows the create, update, and delete functions of CRUD (Create, Read, Update, Delete) within a search and rescue animal shelter database in MongoDB. Austin Animal Center ( AAC) wishes to interact with the database from a user-friendly dashboard. The Pymongo driver will be utilized within Python library to connect to and access the database within MongoDB.

## Motivation

PyMongo allows the retrieval of the data within MongoDB using dictionary-like syntax. Instead of continuously writing queries, single methods can be reused to create and read within the database. AAC wishes to keep the database for animals associated with search and rescue. The database will, in the future, interact with a dashboard to update the animals. CRUD methods within python will allow for easy record keeping of animals within the rescue.

## Getting Started

To start this project, you must have access to the csv dataset of Austin Animal Center Outcomes. The Pymongo driver was chosen because of it’s work with big data and noSQL databases. Since MongoDB is a noSQL database, PyMongo was chosen for ease of interaction with Python.

## Installation

To run, the following must be downloaded:

MongoDB, a Python editor, the PyMongo library, the csv dataset of Austin Animal Center Outcomes.

In this case, we used Jupyter Notebook to create and test the CRUD methods.

The code can be accessed from the following links:

animal\_shelter.py class:

<https://github.com/Emily-Wood7/CS340-Client-Server-Development/blob/3eae3d700f6f18d063c98ea0e68531c19bc54407/Week%206/animal_shelter.py>

testing script:

https://github.com/Emily-Wood7/CS340-Client-Server-Development/blob/3eae3d700f6f18d063c98ea0e68531c19bc54407/Week%206/testing\_script.ipynb

**Usage**

### Code Example

The data first needs to be imported into MongoDB using the **mongoimport** function:  
Graphical user interface, text

Description automatically generated

For future authentication purposes, a user must be created within MongoDB using the **createUser()** function:

Text

Description automatically generated

From this point mongo needs to be stopped then restarted with authentication on:

**mongod\_ctl stop**

**mongod\_ctl start**

In the animal\_shelter class, MongoClient must be imported from pymongo using **from pyMongo import MongoClient**. We also import **bson.objectid** to ensure each item in the collect receives a unique ID.



An object must be created by connecting to the mongo client, the created database needs to be specified, and include the authentication credentials previously created:

A picture containing text

Description automatically generated

Pymongo has built in functions that are equivalent to the function within MongoDB. We are concerned today with the CRUD functions, which include **insert()**, **find\_one()**, **update\_one()**, and **delete()**.

The following code example is from the CREATE and READ operations for CRUD within animal\_shelter.py:

Graphical user interface, text

Description automatically generated

The following code example is from the UPDATE and DELETE operations for CRUD within animal\_shelter.py:

Text

Description automatically generated

### Tests

Within Jupyter Notebook, we import the AnimalShelter class, instantiate the object by passing through our authentication credentials, and create testing variables using the JSON key/value pairs such as an example animal and example query:

Graphical user interface

Description automatically generated with medium confidence

The following tests are for the CREATE and READ functions:

Graphical user interface, text

Description automatically generated

The following tests are for the UPDATE and DELETE functions:

Text

Description automatically generated

The output from the tests should look like this:

Graphical user interface, text, letter

Description automatically generated

## Contact

Emily Wood, emily.wood7@snhu.edu