# CS 340 Animal Shelter Database CRUD Functions

## About the Project/Project Title

This project involves the creation of basic CRUD(create, read, update, delete) functions for a an animal shelter database in MondoDB. The functions will be used in a front-end application so users can edit and update the database as needed. The second part of the project involves an interactive application for the employees of Grazioso Salvare to utilize in screening for search and rescue animals.

## Motivation

This project has been implemented on behalf of the people of the Grazioso Salvare and there search for rescue animals that would make potential search-and-rescue candidates. The CRUD functionality of the database will help optimize their searching and record keeping . The web-application will allow for easy access to records by employees with filters that will help improve their searches for potential search and rescue animals

## Getting Started

To download and run this project, use:

$ git clone <http://THELINKTHEPROJECTISHOSTEDON>

Or, for specific files

$ wget http://THELINKTHISFILEISLOCATEDON

To run the CRUD test files:

$ python3 create\_and\_read.py

$ python3 update\_and\_delete.py

To run the application, open the code in Jupyter Notebook and click “run.”

## Installation

To run these programs you will need to install the following programs:

MondoDB

To install MondoDB, follow the website below and follow the instructions for your respective Operating System(OS)

<https://docs.mongodb.com/guides/server/install/>

Pymongo

To Install pymongo, follow this link as well as the instructions for your respective operating system

<https://pymongo.readthedocs.io/en/stable/installation.html>

For Linux:

$ python -m pip install pymongo

To Upgrade your existing version

$ python -m pip install –upgrade pymongo

Dash Framework

To install the Dash framework on a linux machine, use the following command:

$ pip install dash==1.18.1

Refer to the dash framework websites and tutorials for more information:

<https://dash.plotly.com/installation>

## Usage

The first thing that needs to be done is the database needs to be imported properly to MongoDB. This step can be seen in the screenshot below:

Graphical user interface, text, application

Description automatically generated

Then, you must copy and paste the aac\_crud.py module into your project. Then, you can implement the methods by importing them via “import AnimalShelter from aac\_crud” at the top of your current module. Below is a sample of the aac\_crud.py code being used to modify the AAC animal shelter database.

Upon running the application, you will see an interactive screen that looks like this:

Graphical user interface, text, application, email

Description automatically generated

From there, you will notice a data table that updates in real time to the drop-down menu at the top. This functionality is demonstrated below.

### Code Example

The following screenshot is an example of the aac\_crud.py module code that will be imported to future python code to update the AAC database

Graphical user interface, text, application

Description automatically generated

### The code includes functions for all CRUD functions including creating, reading, updating, and deleting entries from a database.

### Tests

The test “create\_and\_read.ipynb” and “update\_and\_delete.ipynb can be run to demonstrate the functionality of the create and read modules. Below are screenshots of both sections of test code

Graphical user interface, text, application, email

Description automatically generated

This is a sample from “create\_and\_read.ipynb” which imports and implements the functions from the “aac\_crud.py” module. Successful output can be seen below in **Screenshots**.

Graphical user interface, text, application

Description automatically generated

The above picture shows a sample from the “delete\_update\_test.ipynb” file. Successful output of this test code can be seen below in **Screenshots**.

### Screenshots

The screenshot below shows the output of the “create\_read\_test.ipynb” program

Graphical user interface, text

Description automatically generated

Notice how, after running the program several times, the .count() command yields a higher number of entries matching the same parameters in a find() command. That, combined with the successful read() output shows that the create() and read() methods from the aac\_crud.py module imported and executed correctly.

Graphical user interface, text

Description automatically generated

The screenshot above shows a successful deletion of records. Notice how the Object\_id’s in the python cell do not match each other. This means that the delete() method executed and the second read() method of the program is reading a *different* entry.

Graphical user interface, text

Description automatically generated

## 

The next few screenshots demonstrate the Front end application and what it looks like

Graphical user interface, text, application, email

Description automatically generated

The above screenshot shows the data table with the drop-down options as specified by the client. Below is a screenshot that shows proof of the data table and dropdown functionality. Notice how the data table has updated to all dogs (the “Water Rescue” option applies a filter on the data to show specified animals to be screened for water rescue training).

Like the data table, there are interactive applications on the page including a pie chart that shows the combined age of each respective breed in the filter, and a map that pinpoints the location of the first entry on the updated data table.

Graphical user interface, application

Description automatically generated

Notice the differences in this screenshot and the one below it. This provides proof that the applications update in real time.

Graphical user interface, application

Description automatically generated

The final screenshot shows proof of the picture being linked to the website [www.snhu.edu](http://www.snhu.edu) and will redirect once clicked on.

Graphical user interface, website

Description automatically generated

## Roadmap/Features (Optional)

The pie chart currently shows the combined age of each of the respective dog breeds in the filter. While this is a good demonstration of the functionality of the dash framework and all of its utility, however this exact example of data may not be useful to the people of Grazioso Salvare. Maybe comparing the total number of each breed in the filter may be more suitable in the future.

When the application is opened on a larger window, the actual logo is only clickable on a certain portion of the application. Perhaps it may be worth looking into making sure the entire logo is always clickable.

## Contact

Jeffrey Forte