# CS 340 README

## Project Two

## About the Project/Project Title

This project involves the creation of a Python module that enables Create (C) and Read (R) operations, which are part of the broader CRUD (Create, Read, Update, Delete) functionality for interacting with databases. This Python module serves as an interface between a MongoDB database and a client-side user interface. The database in use for this project is the Austin Animal Center (AAC) Outcomes data set. This data set is imported into MongoDB from a CSV file using the MongoDB import tool, and the database used is ‘AAC” and the collections is "animals". The Python module is developed using object-oriented programming (OOP) methodology, with a CRUD class that provides specific functionalities. These include a method called Create which is used for inserting a document into the MongoDB ‘AAC’ database and ‘animals’ collection, a Read method for searching the documents from the MongoDB database and collection, an update method that queries for and changes documents from the MongoDB database and collection, and a delete method that queries and removes documents from the MongoDB and collection. To display the dashboard, I needed to create interactive options to filter the Austin Animal Center Outcomes data set, a data table which dynamically responds to the filtering options, and a geolocation chart and histogram that dynamically responds to the filtering options.

## Motivation

Grazioso Salvare wanted Global Rain to create a software application that can work with existing data from the animal shelters to identify and categorize available dogs. Global Rain has contracted our team for the full stack development of this application, including a database and a client-facing web application dashboard through which users at Grazioso Salvare will access the database. Grazioso Salvare identifies dogs that are good candidates for search-and-rescue training. When trained, these dogs are able to find and help to rescue humans or other animals, often in life-threatening conditions.

**Project Reproduction**

Install MongoDB and PyMongo: Ensure that MongoDB is installed on your system. If not, you can download it from the official MongoDB website. Also, install the PyMongo driver in your Python environment using pip: pip install pymongo.

Setup the MongoDB Database: Open up Terminal, enter the directory address of the saved CSV file($ cd /usr/local/datasets/), Import the AAC Outcomes dataset into your MongoDB instance using the MongoDB import tool while creating the database named "AAC" and a collection named "animals". ( $ mongoimport --username="${MONGO\_USER}" --password="${MONGO\_PASS}" --port=${MONGO\_PORT} --host=${MONGO\_HOST} --db=AAC --collection animals --authenticationDatabase admin --file /usr/local/datasets/aac\_shelter\_outcomes.csv --type csv –headerline)

Setup User Authentication: enter mongosh into terminal, enter use AAC, create the user "aacuser" with a password for the AAC database in MongoDB using ~ db.createUser({user: "aacuser", pwd: passwordPrompt(), roles: [ { role: "readWrite", db: "AAC" },{role: "read", db: "test"}]}).~

Python Module: Write the Python code within the AnimalShelter class for connecting to the MongoClient and then write the code for the Create, Read, Update, and Delete methods.

Dashboard Creation: Import the AnimalShelter class, upload the GraziosoSalvareLogo, create radio call items, create the dash table, create callbacks for the radio items, histogram, and map.

#### **Tools used:**

#### MongoDB server version 6.0.6 – MongoDB is a NoSQL database that provides high performance, high availability, and easy scalability. It works on the concept of collections and documents, allowing for the storage of complex data structures. For this project, it was ideal due to its ability to handle large data sets and its compatibility with PyMongo.

Python 3.10.9: I used VS Code to utilize Python which is the programming language used for this project. Python is widely recognized for its readability, simplicity, and extensive library support. It is especially useful for this project because it offers PyMongo, a Python driver for interfacing with MongoDB.

PyMongo 4.3.3: PyMongo is a Python driver for MongoDB, providing tools to interact with MongoDB databases from Python scripts. It was used in this project to create and read documents in the MongoDB database.

Jupyter Notebook 6.5.4: Jupyter Notebook is an open-source web application that allows the creation and sharing of documents. It was used in this project for testing the Python module and printing the results of the Create and Read operations.

VS Code 1.78.2: Streamlined code editor with extensions for python, MongoDB, and Jupyter Notebook.

CSV Files: The data used for this project is stored in CSV (Comma-Separated Values) format. The MongoDB import tool allows easy import of data from CSV files into a MongoDB database.

MongoDB Import Tool: This is a tool that comes with MongoDB, used to import content from a CSV file into a MongoDB database. It was used to import the Austin Animal Center Outcomes data set into the MongoDB database.

#### **Frameworks and libraries used with Python and Jupyter Notebook:**

dash 2.10.2 – Python framework for building reactive web apps.

pandas 1.5.3 – High-performance, easy to use data structures and data analysis tools.

json – used for converting update and delete results to JSON format

traceback – used for troubleshooting exceptions by printing a traceback to the console

MongoClient – imported from PyMongo to allow queries to the MongoDB database

base64 – allows images to be displayed in the app

dash-leaflet 0.1.23

dash-table 5.0.0 – A first class interactive database for dash

dash-html-components 2.0.0 – Dash UI html component suite

dash-core-components 2.0.0 – Dash UI core component suite

Jupyter-dash 0.4.2 – Dash support for the jupyter notebook interface

plotly-express 0.4.0

**Running Dash:**

**A screenshot of a computer

Description automatically generated**

**Working Radio buttons:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Challenges**

Some of the errors that I ran into required that I upgrade the frameworks or libraries to ensure compatibility since I ran the project on my own computer not the Apporto server the school provides. Once I had the project running I encountered a KeyError for the histogram which after searching online I was easily able to fix. Apart from that error, I only received syntax error warnings and I fixed those quickly.

## Contact

James Soto