# CS 340 README

## About the Project/Project Title

This project is a basic CRUD python module that connects to a MongoDB database with information regarding the Austin Animal Shelter’s data. This application will be able to create new records and read them from the entire dataset stored in the MongoDB database. The goal for this application is to have an executable python script that can update the MongoDB system without manually logging in and adding the information directly into the program. The CRUD module is what will allow the dashboard to communicate with the database. It gives the application a way to create new entries, read the data back to the user, update those entries, and delete those entries.  
 The CRUD module will serve as the database for an interactive dashboard for a web application. The dashboard relies on the CRUD module to retrieve and filter through the data based on user input. A user can select a preset button to filter through different datasets that is relevant to the business owner. The dashboard also features widgets like data visualization through a pie chart and instant geolocation information.

## Motivation

The Austin Animal Shelter deals with a lot of clients coming in and out of the center. This means it is essential to have an efficient program to keep up the demand of the business daily. This program helps achieve the goal of streamlining data input and retrieval so that the associates have more time to focus on what matters most, taking care of the animals.

## Getting Started

To get a local copy up and running, you will need an existing CSV dataset with specific data about each instance that pertains to your business. You will also need a MongoDB database with outcome data that matches your CSV file’s instances and attributes, and a Python environment that can run the module. In this example, we are importing the AAC database into MongoDB before loading the mongo shell.

1. Import the CSV dataset into MongoDB.

In the terminal in Codio, change into the datasets directory and run:

mongoimport --type=csv --headerline --db aac --collection animals –drop ./aac\_shelter\_outcomes.csv

This will load the AAC data into the aac.animals collection.

1. Load the Mongo shell and create a user account

Open mongosh and access the admin database. Create a user with read/write access to the aac database.

db.createUser({ user: "aacuser", pwd: passwordPrompt(), roles: [ { role: "readWrite", db: "aac" } ] })

1. Place the CRUD module and notebook together

Ensure the CRUD Python file is in the same directory as your Jupyter Notebook. Import the class using:

from CRUD\_Python\_Module import AnimalShelter

Next, create an instance, then use the create(), read(), update(), and delete() methods to work the data.

1. Code and test the Dashboard

Open the Jupyter Notebook in Codio and ensure that all required Python 3 libraries are available. Import the data using via the AnimalShelter class whicg handles the communication to the MongoDB database. Use Dash and HTML to design the webpage. This should include a design image, title, button functionalities, table design, and visual widget design. Run the notebook to test the dashboard and all components work correctly.

## Installation

This module uses Codio Learning Environment that provides both Jupyter Lab for Python development and a Linux terminal for MongoDB access. We use the Python 3 language for the writing and testing scripts, and PyMongo is used for communicating with the MongoDB database. All these tools are included within Codio so there is no installation needed locally. To work with MongoDB within Linux, you use mongosh in the terminal. If you needed to install PyMongo in your local IDE, you would use pip install PyMongo within your Python environments and install MongoDB into your local device from the main website. In this project we also utilize BSON to transmit records over the network because BSON is utilized for MongoDB instances.

## Usage

This module uses PyMongo as the Python driver so that the users and developers can communicate directly with MongoDB without actually running commands within the MongoDB database through the Linux terminal. Each CRUD function within the application is mapped though the Python script as follows:

* Create() inserts a new document in the database.
* Read() returns a list of documents that match the query.
* Update() changes fields within matching documents of a query.
* Delete() removes the selected documents that match a query.

This is an example of how this module will work in Python.  
  
Inserting a new document into the database:

A screenshot of a computer program

AI-generated content may be incorrect.

Then you can read documents that match a specific input:  
A screen shot of a computer code

AI-generated content may be incorrect.

### You can update documents that are queried: A screenshot of a computer program AI-generated content may be incorrect.

And you can delete documents that are queried:  
A screen shot of a computer program

AI-generated content may be incorrect.

These CRUD methods are also used by the interactive dashboard. The dashboard imports the AnimalShelter class and uses the read() method to retrieve data from MongoDB. When a filter is selected, a query is passed through CRUD and the returned data is displayed in the components.

### Code Example

Here is how the core CRUD methods were implemented within the module:

*A screenshot of a computer code

AI-generated content may be incorrect.*

### A computer screen shot of a program code AI-generated content may be incorrect.

A computer screen shot of a computer code

AI-generated content may be incorrect.

Here is how the Dashboard was designed and implemented:  
A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.  
A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

A computer code with text

AI-generated content may be incorrect.

### Tests

Testing for the CRUD module was done in a Jupyter Notebook. First I imported the module and created an instance of AnimalShelter. Next I called the create() function with a new record to insert a test animal. Afterwards I called the read() function with a sample query ({"name": "Tucker"} to test that the communication between the Python scripts and the MongoDB databases were communicating using PyMongo. Next, I expanded the test script to call the update() and delete() functions. The update() function works to update a specific document and changed one field to verify that it was modified. I was able to test that the delete() function worked as designed by selecting the documents unique ID for deletion. I also added read() functions in between each step to verify that the update() and the delete() functions worked. I ran the program and verified that the testing resulted in a successful build. I have attached the successful build and output below with screenshots.

After successfully verifying the CRUD module was working as expected, the dashboard is tested by running the code and ensuring all dashboard components like the image, texts, and widgets displayed correctly. Also, each radio button was tested to make sure it updated the table, pie and geographic location. This testing validated the Dash callback functions used in the controller blocks and that the update\_dashboard() function worked to filter MongoDB data. I also validated the update\_map function for updating the geolocation map when a button or row in the table is selected. During testing, I had an issue with debugging the pie chart because it initially displayed every breed in the dataset when the “reset” button was selected. This caused the pie chart to render poorly since it had hundreds of breeds populating the chart, causing a “shadow” effect. That issue was resolved through refining the code, so the chart only shows the most relevant breeds, which made it appear much cleaner and filtered the noise. Screenshots of the dashboard working are included below.

### Screenshots

*A screenshot of a computer program

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.*

*A screen shot of a computer

AI-generated content may be incorrect.*

A computer code on a white background

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

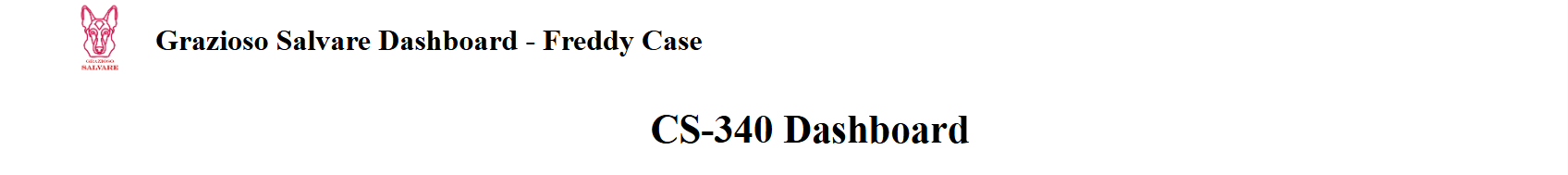
*A screenshot of a computer screen

AI-generated content may be incorrect.  
A close up of a text

AI-generated content may be incorrect.*

*A computer code with many black and white text

AI-generated content may be incorrect.  
  
Dashboard:*

**

*Reset button:*

*A screenshot of a computer

AI-generated content may be incorrect.*

*A pie chart with numbers and symbols

AI-generated content may be incorrect.*

*A map with a location pin

AI-generated content may be incorrect.*

*Water Rescue:  
A screenshot of a computer

AI-generated content may be incorrect.*

*A screenshot of a computer

AI-generated content may be incorrect.*

*A blue circle with white text

AI-generated content may be incorrect.*

*A map with a location pin

AI-generated content may be incorrect.*

*Mountain or wilderness Rescue:*

*A screenshot of a computer

AI-generated content may be incorrect.*

*A screenshot of a computer

AI-generated content may be incorrect.*

*A pie chart with different colors

AI-generated content may be incorrect.*

*A map with a location pin

AI-generated content may be incorrect.*

*Disaster Rescue:  
A screenshot of a computer

AI-generated content may be incorrect.*

*A screenshot of a computer

AI-generated content may be incorrect.*

*A pie chart with numbers and a few percentages

AI-generated content may be incorrect.*

*A map with a location pin

AI-generated content may be incorrect.*

## Contact

Freddy Case