# CS 340 – Project Two README

**Clay Dodson**

**06/26/2022**

## About the Project/Project Title:

I call this project the Animal Center Database. This project is a database for storing information for a rescue-animal training company including age, breed, and other attributes. The project also contains a simple CRUD (create, read, update, delete) program which allows a user to update the database as needed.

## Motivation:

This project is being developed in order to make the organization of the data for the Grazioso Salvare company easier to access and update. Once the entire project is built out, authorized users will be able to create, read, update, and delete information within the database.

## Getting Started:

To get a local copy up and running, follow these simple example steps (note: you will need to use Linux and have MongoDB and Jupyter Notebook already installed on your computer):

1. Using the Linux command line interface, login to MongoDB using your localhost port number.
2. Add the Austin Animal Center (AAC) database and its affiliated documents.
3. Create indexes in MongoDB to increase efficiencies of queries.
4. Create user authentication in MongoDB for an admin role and user role. Each role will have a unique password to access the database.
5. Run queries and update documents within the AAC database to confirm proper access to the database.
6. Develop code in Python to create, read, update, and delete the documents within the database.
7. Create Jupyter Notebook file (IPYNB) with correct layout, functions, and callbacks to display the user dashboard.
8. Use Dash/Leaflet to create the geolocation map and graph which displays the data to the user.

## Installation/Tools:

For this project, you will need to have Linux installed on your system along with MongoDB and Jupyter Notebook. For Linux, you will need to use a bootable external drive to install it if your system doesn’t already run Linux. MongoDB can be installed from mongodb.com and Jupyter Notebook can be installed from jupyter.org.

MongoDB and Python are used for this project because they work well together when dealing with JSON/BSON documents. Since MongoDB is a NoSQL database then it provides a lot of options for scaling projects and setting up queries. The Dash framework is used within Python which allows you to create simple user interfaces. Dash allows one to add graphs, maps, buttons, and other functionality in order to display information from a database and an user-friendly manner.

There were several steps taken in order to get the project functional. The first step was creating a CRUD module in Python which acts as the driver for the program. Then, you must create a Jupyter Notebook file which uses the CRUD Python module to access and display the data from the Grazioso Salvare database. Finally, you must use the Dash framework to build out an interface/dashboard for the user to access and display the data in a manner that fits their needs.

Unfortunately, I hit a hard roadblock when trying to add the geolocation code to the project. I was able to get the dashboard up and running but when it came time to add the geolocation map and callbacks, I kept running into syntax errors that I couldn’t resolve. I reviewed documentation for MongoDB, PyMongo, Dash, Leaflet, and StackOverflow posts but couldn’t resolve my issue. Therefore, I had to turn in an incomplete project since I was up against the deadline. While this term was unique for me due to a death in the family, I will be sure and plan better for these larger projects in the future to make sure I have time to resolve these issues when I hit a roadblock.

**Resource Links:**

[**https://www.mongodb.com/docs/manual/**](https://www.mongodb.com/docs/manual/)

[**https://api.mongodb.com/python/2.3/faq.html**](https://api.mongodb.com/python/2.3/faq.html)

[**https://dash.plotly.com/dash-core-components**](https://dash.plotly.com/dash-core-components)

[**https://dash-leaflet.herokuapp.com/**](https://dash-leaflet.herokuapp.com/)