# CS 340 README Project Two

## About the Project/Project Title

This project is a tool that can take a database of shelter animals and turn it into a user-friendly dashboard that can quickly show potential matches in the area. Grazioso Salvare engaged us to make this dashboard and make it open-source and available for all to use. They train search and rescue dogs; this project was designed to make that search much simpler. It could be used with any database of shelter animals, but for this project we are using the Austin Animal Center’s data to identify potential candidates.

## Motivation

Searching through a database of over ten thousand rescue animals can be really dauting, especially when the animals you are looking for need to match several attributes. A database specialist could query the data as needed, but what about the rest of us? The goal of this project was to make a dashboard easy enough so that anyone could use it to find what they need, with easy-to-follow maps, pie charts, and sortable tables. A few buttons were added to give Grazioso Salvare quick answers to a specific set of queries; animals that match attributes to make them good candidates for several different types of rescue operations.

## Getting Started

To begin, you will need to implement the database. This begins in Powershell with the following command:

A screen shot of a computer screen

Description automatically generated

This will populate our database from a csv file. On success, it should look like this:

A computer screen with white text

Description automatically generated

We will need to create a user account to interact with the data. Make sure to keep up with the user and password, as we will need to pass that through our middleware to access our data:

A screenshot of a computer

Description automatically generated

Note that even through we are creating a user for the AAC database, we need to do that from within the “admin” database.

From here, you simply need to create a folder with the attached CRUD module in it, along with the attached dashboard script. This code will do the rest of the work. You will need to modify the username and password fields in both files to match the username and password you created in the step above. The simplest way to do this would be to open the file in a text editor (such as Notepad), make the changes, and save the file.

## Installation

You will need:

Python (download from python.org)

Python is a scripting language; there are several choices for this, but Python is the preferred language of MongoDB.

MongoDB (download from [www.mongodb.com](http://www.mongodb.com))

MongoDB is a NoSQL database used for many online applications.

MongoClient from the pymongo library (can be added via pip install after python installation)

The MongoClient feature from the pymongo library gives us the tools to connect our module to our running MongoDB database so that we can pass commands to interact with documents.

Several additional python modules: Dash, plotly, pandas; these are necessary for dashboard implementation. Dash gives us elements for our page, while plotly and pandas give use the tools we need for maps and graphs and other visualizations.

## Usage

This application makes use of a MongoDB database to store data, a Python middleware layer utilizing pymongo, and a Dash front end framework implemented in Python. Mongo is a free, open-source NoSQL database solution that works perfectly for a lightweight project like this one. It is also the recommended database for Python projects; the two work very well together through pymongo. Dash works well with both Python and Mongo, but also with our plotly and pandas modules for a nice and easy implementation. All of this leaves us with an easy-to-use dashboard that works as follows:

**Screenshots:**

We begin at our base page, which starts with unfiltered data:

**A screenshot of a computer

Description automatically generated**

Clicking on the various radio buttons at the top of the dashboard will give us some pre-sorted results. This is a code snippet and the results for Water Rescue:

A screen shot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

And here we have a snippet and results for Mountain/Wilderness Rescue:

A computer screen shot of text

Description automatically generated

A screenshot of a computer

Description automatically generated

Then we have a page for Bloodhound-type dogs used for disaster rescue and missing persons:

A screen shot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

And finally, we have a button to reset the table and visualizations back to the original view:



A screenshot of a computer

Description automatically generated

**Challenges**

This was my first project of this type, so I had to learn a lot as I went along. I used a lot of tools to check and recheck items to make sure things were going according to plan. I had to modify my original middleware file to correct some issues I had with importing data; the original data would load, but I couldn’t get the data to update when pressing buttons. The error codes helped, but a couple of them were several hundred lines long. It wasn’t too hard to follow the flow of data through the program, but in places where the modules abstracted the data away it was much more difficult to follow. And at least one time, I modified some code and somehow had an extra space or something in it; I looked over it at least a dozen times before deleting the lines and rewriting them, which took care of the problem I couldn’t see.

## Contact

Your name: Buddy Marcey, buddy.marcey@snhu.edu