# About the Project

This project creates a website that displays data based on the Austin Animal Center data set. This website is interactive, and allows for filtering of different rescue types, and even displays the data on a pie chart/geolocation chart!

# Purpose & Motivation

The motivation behind this project was to create a data set that could be filtered to match certain types of dogs. Specifically rescue based dogs, things like Water, Mountain, Wilderness, and even Disaster or Individual Tracking. The reason this is so important is that these dogs can be found more quickly and consistently.

# Dependencies

This software requires python and MongoDB to run correctly. It also uses the Dash framework as the main library tool for running the website. This is very useful as it provides a view and controller structure for the application.

MongoDB: https://www.mongodb.com

Python: https://www.python.org/downloads/

The purpose of MongoDB in this project is to easily interact with the Austin Animal Center data set. Python is being used here with MongoDB, as it's very optimal for quick front end websites.

# Steps taken

Below is a list of the steps taken to complete this entire project. Overall, it was a pretty fun project, and I learned quite a bit about MongoDB and handling websites with Python.

\* Created an aacuser and aac database

\* Imported the Austin Animal Center into the database

\* Created a python script that handled basic CRUD operations

\* Got a basic version of a Dash website working

\* Added the logo and the title

\* Got the data to display correctly on the website (Table/Graphs)

\* Created interactive features for the data, such as filtering for rescue types

\* Created mongoDB queries that made these features work

\* Took a proud look at all my hard work

# Challenges Faced

The below challenges faced were the bigger ones, but I faced a lot of tiny challenges a long the way, as with most software I've written. Most things I fixed with a lot of print statements and googling.

##### Data wasn't loaded

To start, initially all my data wasn't even loaded correctly into the right database. I kept running and running my code wondering what was happening. I eventually solved this problem through print statements, and I realized looking back at my old screen shot, that I never imported the data correctly via the import command. I thought I did because of the success message, but it was never going to the right place.

##### Read function wasn't correct

I realized this was an issue after comparing the prints from what my function was returning, and what was being shown in an example. After a while I realized I just needed to update the read function, which was a simple thing to fix.

##### Website wouldn't load

This one was a nightmare because, even after all the documentation I read of creating the website, loading the data, and running the software, nothing would happen. This one made me lose my mind a little bit. I still don't fully know why it wouldn't work, but I solved the problem by moving away from just using JuptyterDash to using pure Dash.

##### Mongo Queries

This one was confusing at first, but once I scrolled down in the documentation, I realized there were requirements per filter type. It became clear I didn't know how to do that kind of query off the top of my head, so I googled "operator queries mongodb" as it seemed logical as there were multiple requirements per query, and I thought maybe it would work. I learned about the $or and $and operators, and after testing this became simple. My biggest concern is the efficiency of it, the query is quite large, and nothing is async, I feel that this probably isn't a very efficient website.

# Functional operations