# CS 340 README Template

## About the Project/Project Title

This program allows a user to access the AAC database via a web application. This application allows the user to look up all animals within the database and then filter the results down to see animals that meet the specifications for different rescue types.

## Motivation

This application used a couple tools that were specifically chosen because of the nature of this program and the ease of use. MongoDB was chosen to be used for this project because it provided a way to quickly form a database from the CSV file given, as well as having the capability of interfacing with Python very easily. Although Python could be used with other database tools, to be able to run simple CRUD functions would be more complex within an SQL database compared to what it was using MongoDB with Python.

The project also used Dash in the construction of the dashboard. This tool was chosen because of the dynamic nature of the framework. Like MongoDB, Dash provides an easy interface with Python. It allowed for the use of UI elements like the graphs and radio buttons to be used right with the Python code.

## Getting Started

To get started there are several steps:

1. Set up the Mongo database by importing the aac\_shelter\_outcome.csv file into the database.
2. Create a user account that has readWrite access to the AAC database
3. Install and access Jupyter Notebook to use the Python Module and Dashboard scripts.
4. Run the Dashboard Script to start the web application. The Script will connect to the database using the username and password set up in step 2.

## Installation

1. Current version of Jupyter Notebook – Follow instructions on https://jupyter.org/install
2. MongoDB – Follow instructions on https://www.mongodb.com/docs/manual/administration/install-community/
3. Python – Follow the appropriate instructions on https://realpython.com/installing-python/

## Usage

The application has three different functions to it. The first is to provide a data frame that initially returns all animals in the database, but then has filtering options to return specified animals for different rescue types. This is done by clicking one of the radio buttons which will run a database query and return the results in an updated data frame. The next function of the application is the bar graph, which displays the break down of the breeds in the data frame (Just what is showing, not the whole database). The last function is the map which dynamically updates marker for the location of the animal that was last selected via the checkmark option on the data frame.

### Screenshots

Initial start of the dashboard:

A screenshot of a computer

Description automatically generated

Water Rescue selected:

A screenshot of a computer

Description automatically generated

Mountain/Wilderness Rescue:

A screenshot of a computer

Description automatically generated

Disaster Rescue and Individual Tracking

A screenshot of a computer

Description automatically generated

**Challenges:**

I really didn’t have many challenges that I can think of. One that took me a little bit was the code base that was provided had the initial import statement:

**from** jupyter\_plotly\_dash **import** JupyterDash

This was giving me issues and not allowing the application to run. After looking this up and researching a little I changed it to:

**from** jupyter\_dash **import** JupyterDash

This then allowed the application to run as expected.

## Contact

Your name: Thomas Bartlett