# Animal Rescue Recruiter Dashboard Project 2 CS 340 README

## About the Project/Project Title

Interface creation of Search and Rescue Dog Recruiting Data Filtering:

A user interface is being created to simplify the process of interacting with a large database. This simplification includes the creation of quick filtering that can help the users find what they are looking for quicker than it would be to navigate the MongoDB database.

## Motivation

Many of the dog shelters will have their own way of organizing data and compiling them can be an issue. If we can upload the data into a single database and help standardize the data so most of them have them same and consistent data, it will be easier to look through multiple organizations data all at once. A specific interface for the organization compiling all this data will be created as well to help simplify this process further.

## Installation

Prior to this README, instructions were written on how to connect Python to MongoDB and use python to interact with the database. This installation process will provide guidance on writing Python Code to help set up for the Dash interface with the use of the Dash Layout code guidance and the callback system that Dash uses. These general instructions for [Dash](https://dash.plotly.com/introduction) can be found by clicking the link.

## Getting Started

Follow these steps to create the Dash Interface:

1. Start MongoDB by initializing it using: /usr/local/bin/mongod\_ctl start
2. Import all necessary components and libraries as show below:  
   Text

   Description automatically generated
3. Use Python to login to MongoDB and read the data to a dataframe in Python:  
   df = pd.DataFrame.from\_records(shelter.read({}))
4. Set up the layout of the JupyterDash App layout. See below for an example. You can see items that are highlight such as where the image goes, where the dropdown menu goes, and the datatable.   
   Text

   Description automatically generated
5. Configure Callbacks to use these items with Input and Output that link to the id’s of the items in the app layout.   
   Text

   Description automatically generated with low confidence
6. Test the Dash App Interface and Interactivity to ensure all widgets are properly functioning.

## Usage

The dash application interface built for the filtering system is centered around a default table of the data that is pulled from MongoDB. Users can search for specific items in the table by text by filling in the top of the graph. If users need to filter quickly for specific kinds of breeds suited for certain rescue work, users can now use a drop down to jump to those queries quickly. Two widgets have been installed to help with the search. The first widget is used to locate where an animal may be on the map. The second widget is to chart out how many of the different breed types there are in their specific query. The application automatically updates so when changes are made to the drop down, all widgets and tables react.

### Screenshots

**Homepage which lists the default table and no selection of the dropdown.**

Graphical user interface, application, table

Description automatically generated

**Filtered List for Water Rescue**

**Graphical user interface, text, application

Description automatically generated**

**Mountain or Wilderness Rescue**

**Table

Description automatically generated**

## Disaster or Individual Tracking

Graphical user interface, text, application, email

Description automatically generated

## Reset Dropdown Filter

\*\*\*Returns the table to default.

Graphical user interface, text, application, table

Description automatically generated

**Widgets**

Based on the current filter selected, the geo map locates the animal at the top of the list and a bar graph will show the breakdown of dogs in the current filter based on breed. Two screen shots below will show the default view of these widgets and the updated one once the ‘Mountain or Wilderness Rescue’ filter is applied.

**Default View**

**Graphical user interface, application

Description automatically generated**

**Chart

Description automatically generatedWidget Updates once the Mountain Or Wilderness Rescue Filter is applied**

**Challenges**

During this project, it wasn’t super clear how to set up all the code. The instructions and articles on [Dash Core Components](https://dash.plotly.com/dash-core-components) was a great help to really find all the information needed to get the code down properly. The callbacks were also a bit difficult. Tracking logic was much of the issue at times to figure out the best order to do things. Some items were being updated before others so tracking that order helped to make decisions on callbacks. I also had to turn to resources such as forums where some other programmers and developers were also trying to figure out the code. There are many resources online and they aren’t exactly the same so pinpointing a code that is familiar and practicing by reading that code and understanding it helped me work through my code in this project.

## For any questions, please contact:

Enrique Zarate

Enrique.zarate@snhu.edu