# CS 340 Project 2

## About the Project/Project Title

*This project is a dashboard for Grazioso Salvare who specializes in search and rescue dog training.*

## Motivation

*The goal of this project was to build a streamlined user-friendly dashboard where Grazioso Salvare could easily narrow down the type of dogs they were looking for.*

## Required functionality

*The required functionality is to have the ability to filter between three categories of potential dogs: Water Rescue, Mountain or Wilderness Rescue, and Disaster Rescue or Individual Tracking. In addition, a way to reset back to non-filtered data. In addition, there is a chart and a geo location map that is required to dynamically respond to these filtering options.*

*When first opening the dashboard, the view is nonfiltered. The map shows the location of the first dog, the breed on hover over the marker, and the name on click of the marker if one is present. The pie chart shows the different animal types. When the view is filtered via the drop down, the map changes to each time to show the first dog on the newly filtered list. The pie chart changes with each filter and shows the different breeds from the filtered data.*

## Tools Used

*I am using PyMongo driver to create the CRUD functionality of this application. This driver allows easy access to the Mongo database and the simplicity to create a CRUD application, it is also well supported and widely used. MongoDB allows for seamless use with python, dash, and leaflet. These components and framework give you the ability to create a detailed view of your data and have a ton of built-in features such as charts and maps.*

*Dash is the framework that provides the overall view and controller for the web application. The control logic is built with the view so that it is integrated into one application. With dash you have the layout of your app in which you use items to create call backs. Call backs are the functionality and controller structure of your application. Call backs are designed to respond to clicks or other input which initiates the call back. This functionality of the call back allows you to dynamically change your application based on input from the user.*

***Links:***

*Dash:* [*https://dash.plotly.com/introduction*](https://dash.plotly.com/introduction)

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

*Jupyter:* [*https://jupyter.org/*](https://jupyter.org/)

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

***Steps taken to complete project***

*The first thing that was developed was the Python crud module. Next was the Juypter notebook file in which we run the application. To begin this process, the first thing to do was to read over documentation for the frameworks going to be used like dash and leaflet.*

*Once connected to my MongoDB instance through the crud application, I started off with an unfiltered view of the data. Next was the creation of the geo location map to show the animal that was listed in the first row of the data. After this, the drop down was developed to allow the required filtering functionality. Lastly the pie chart was developed to dynamically change based on the filtered data.*

***Challenges***

*There were definitely some challenges with developing this dashboard and using the dash framework for the first time. One overall challenge was just getting used to how the dash framework is used and understanding components such as the call backs. To overcome this challenge, I relied heavily on the documentation and trial and error until I understood how everything fit together. Another challenge was trying to use the “dash.callback\_context” to determine the amount of clicks on a button. I implemented this function as laid out in the documentation, but I could not get it to work correctly. To overcome this challenge, I instead developed the filtering method through a drop down instead of buttons. In hindsight, the drop down was much easier to use, and the user could now see what it is they filtered for instead of trying to remember what button they clicked last.*

## Contact

Seth Reeise