# CS 340 README Web Application

## About the Project

This project is Python web application that allows users to search through a database, via a dashboard, to better visualize the data as well as identify and categorize database members.

## Motivation

This project was an assignment for my client/server development class. In completing this project, I will have gained experience developing client/server applications that interface client-side code with databases. The project’s development was divided into three sections: MongoDB Authentication and Indexing, Python Middle-ware CRUD functionality, and the development of the Python Web Application.

## Getting Started

This project uses MongoDB, Jupyter Notebook, and the Dash framework. It has been fully developed to produce a web application that can be used to search through a database and visualize the data via a geo-location map. You can make use of this application by running the ipynb file in Jupyter Notebook.

## Installation

Currently, this project uses [MongoDB](https://www.mongodb.com/) to access and query databases, and [Jupyter Notebook](https://jupyter.org/) to implement a Python module that enables CRUD functionality. [Dash](https://plotly.com/) was used to create the data application in Python. Dash is the leading low-code platform for data applications.

**Create:** The C in CRUD was developed using a create method. Data is passed to the method as a dictionary, then inserted into the database. The method returns true if data was created successfully, and false if empty data was passed into the method.

**Read:** The R in CRUD was developed using two methonds: read and read\_all. Data is passed to both methods as a dictionary, then queried within the database using find\_one(data) or find(data). Read returns the query result, and read\_all returns a cursor.

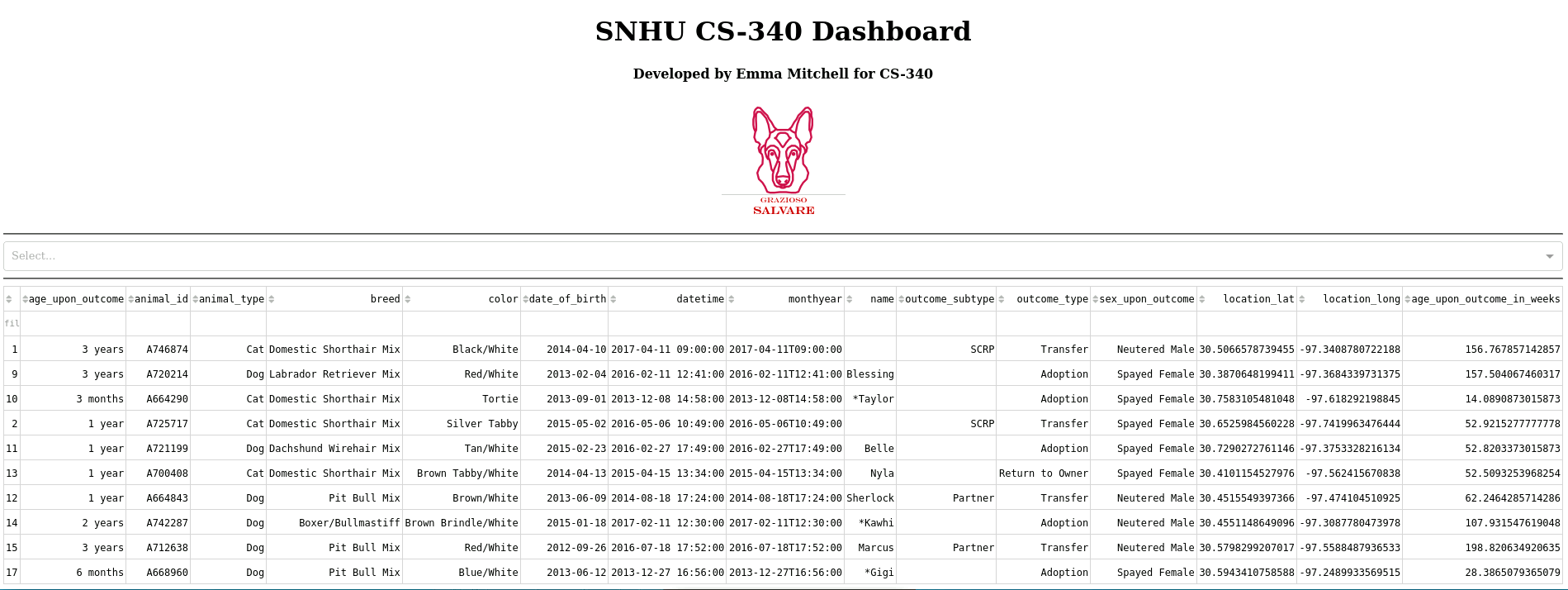
**Update**: The U in CRUD was developed using an update method. Data is passed to the method as two dictionary values: the query, and the updated data. The result is returned if successful, and a MongoDB error message is returned otherwise.

**Delete**: The D in CRUD was developed using an update method. Data is passed to the method as a dictionary, then queried and removed from the database using remove(data). The result is returned if successful, and a MongoDB error message is returned otherwise.

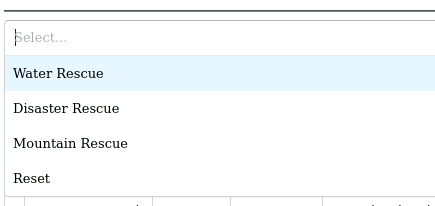
## Usage

### Code Example

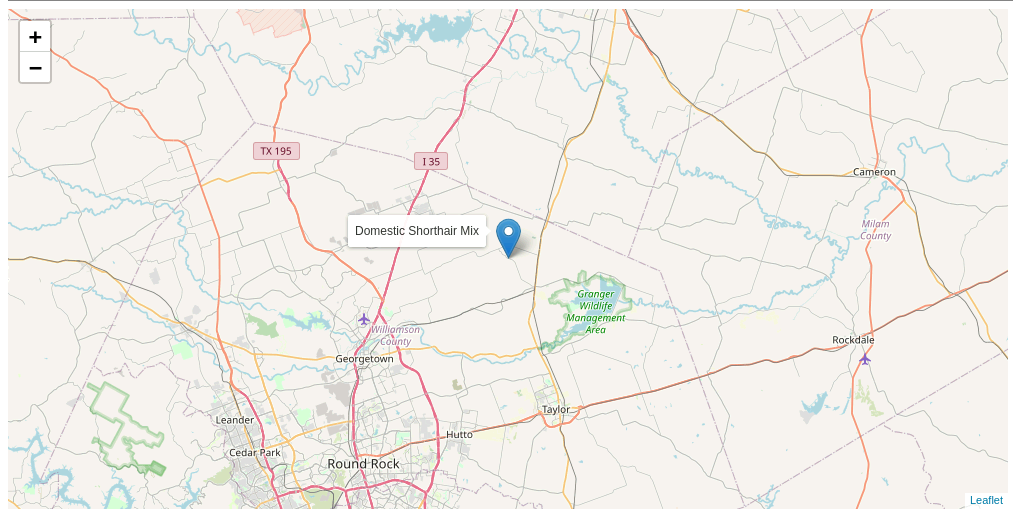
Using MongoDB, the Python module in the PY file, and the dashboard ipynb file, you can access and make use of the web application. Here is an example of what the application looks like.



As requested, the application allows for filtering results based upon three categories: Water Rescue, Disaster Rescue, and Mountain Rescue. There is also an option to reset to an unfiltered view of the database. You can select a filtering option via the dropdown as seen below.

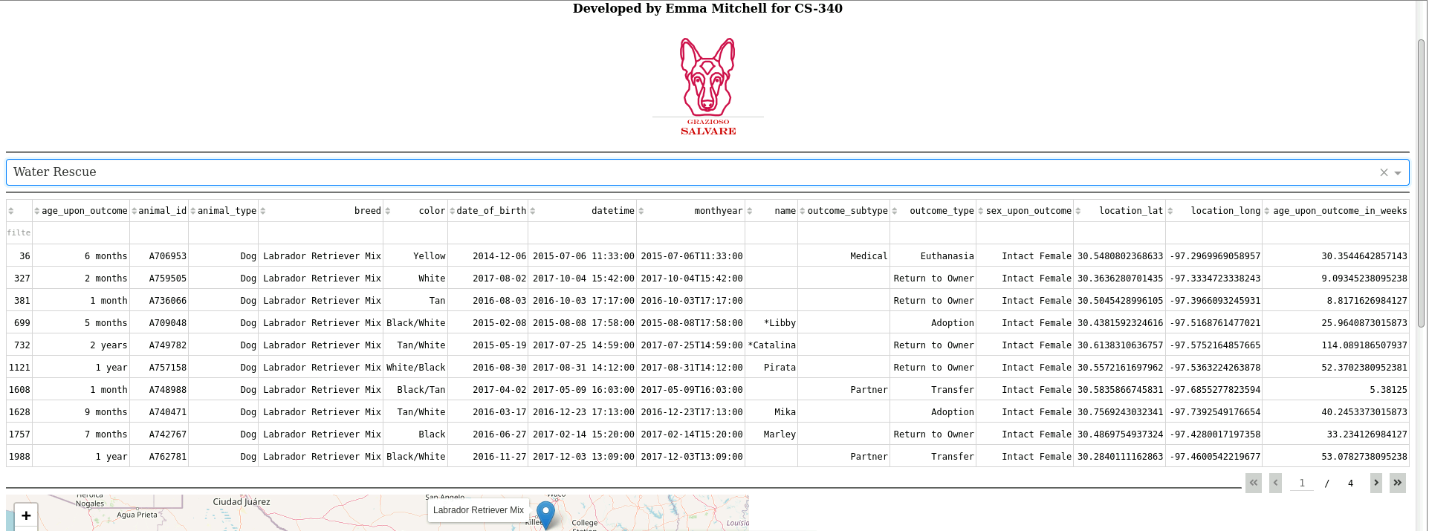


Finally, the web application has a geo-location map that shows the location of the first animal within the data table results. This map updates when the database is filtered. Below is an example of the map and its output.

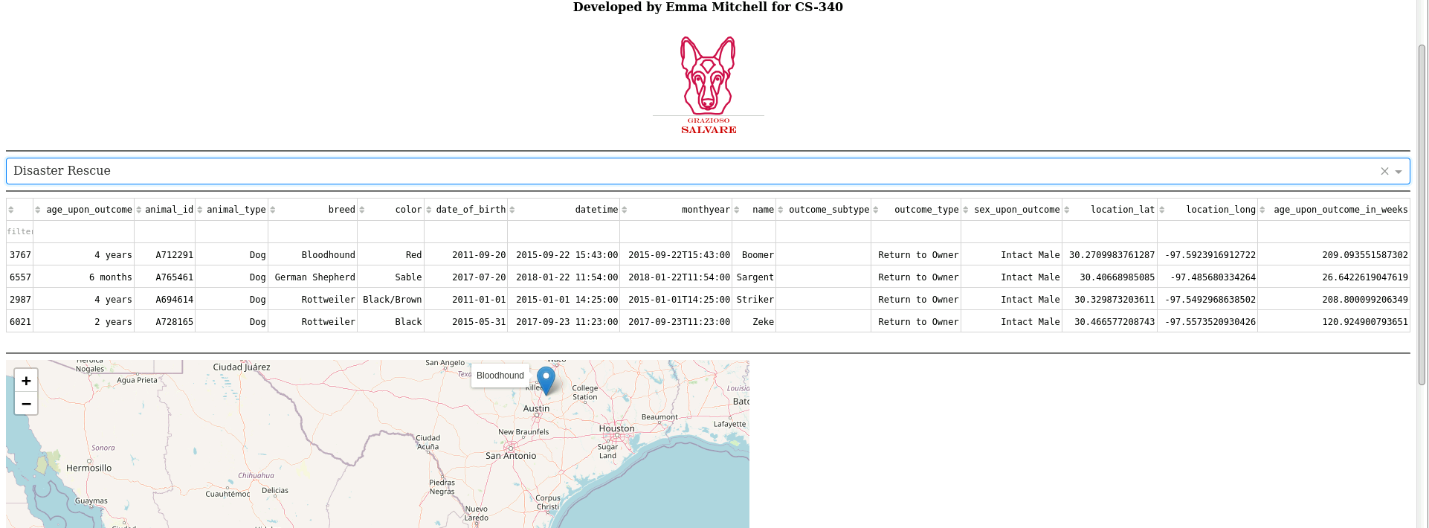
**Tests**

The following screenshot document how the data table and map update when each of the filtering options are selected. Take notice of the results and the name on the map for each screenshot.

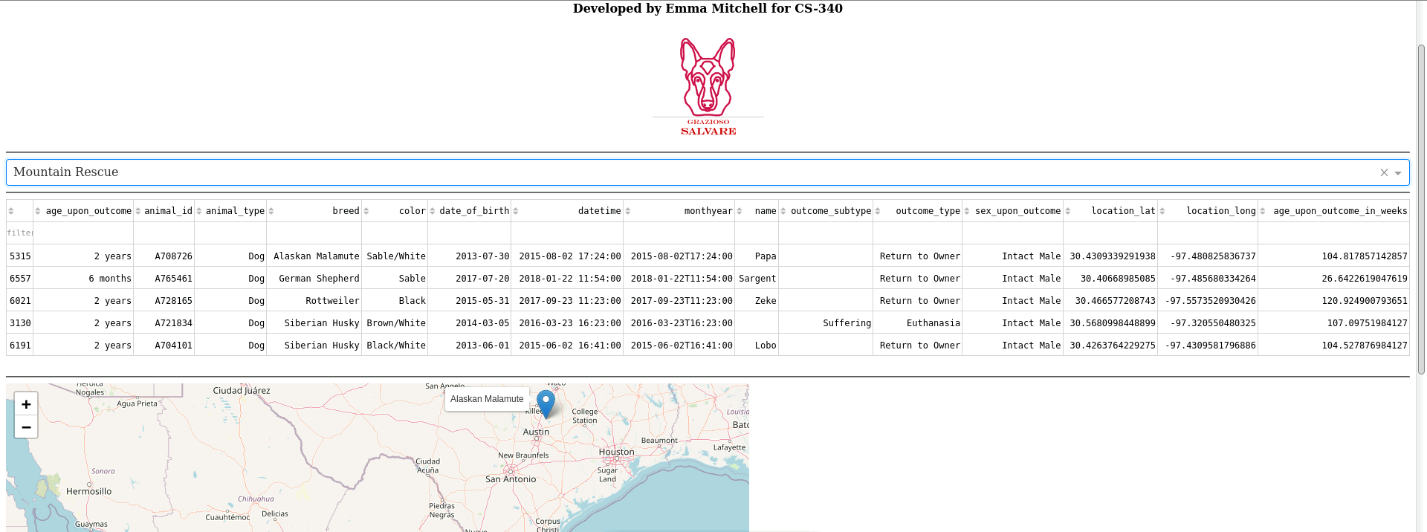
**Water Rescue:**



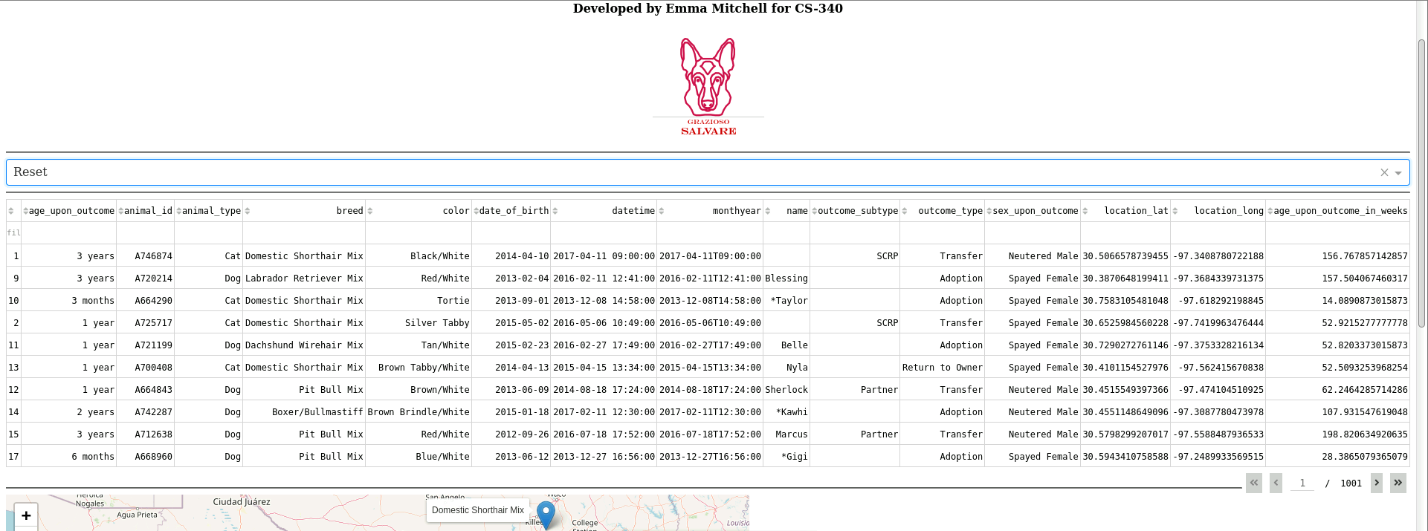
**Disaster Rescue:**



**Mountain Rescue:**

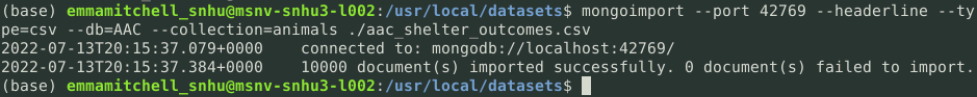


**Reset:**

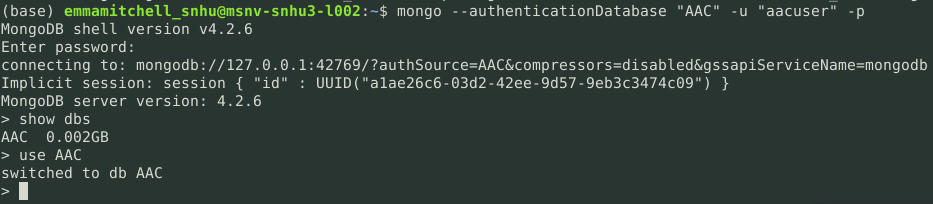


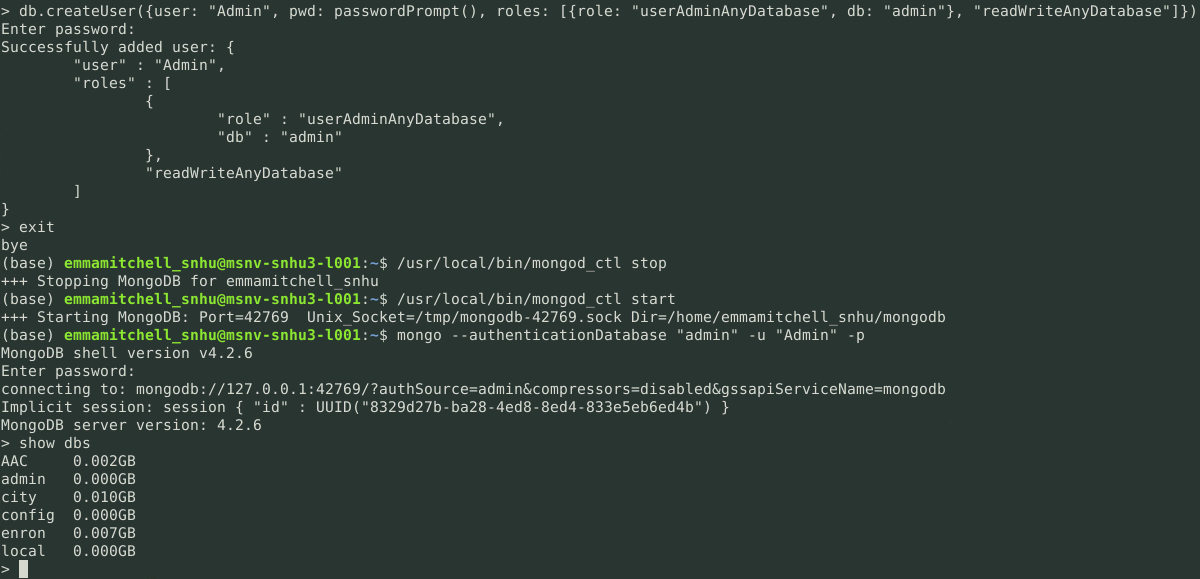
### Additional Screenshots

Here is a screenshot of the MongoDB import execution.



Here are screenshots of user authentication execution.





## Contact

Your name: Emma Mitchell [Last Updated 8/14/2022]