# CS 340 Project Two README

Mark Baligad

## About the Project/Project Title

The Grazioso Salvare Candidate Search dashboard facilitates searching for candidate animals to serve as rescue animals. The interactive dashboard enables users to filter based on animal type, breed, and other features, view a detailed data table listing each animal, review a chart displaying the breed distribution, and view the location of the animal on a map.

## Motivation

The international rescue-animal training company Grazioso Salvare identifies dogs that are good candidates for search-and-rescue training. This application stores and categorizes data from animal shelters to help Grazioso Salvare identify candidates to train with the potential to excel in life-saving rescue operations. The Global Rain software engineering company is proud to support Grazioso Salvare and the nonprofit animal shelter agencies in this effort.

## Getting Started

If you want to get started working with the Grazioso Salvare Candidate Search application, clone this repository and follow along with the installation steps below. See the Sample Screenshots section for dashboard filter examples.

## Tech/framework used

Built with

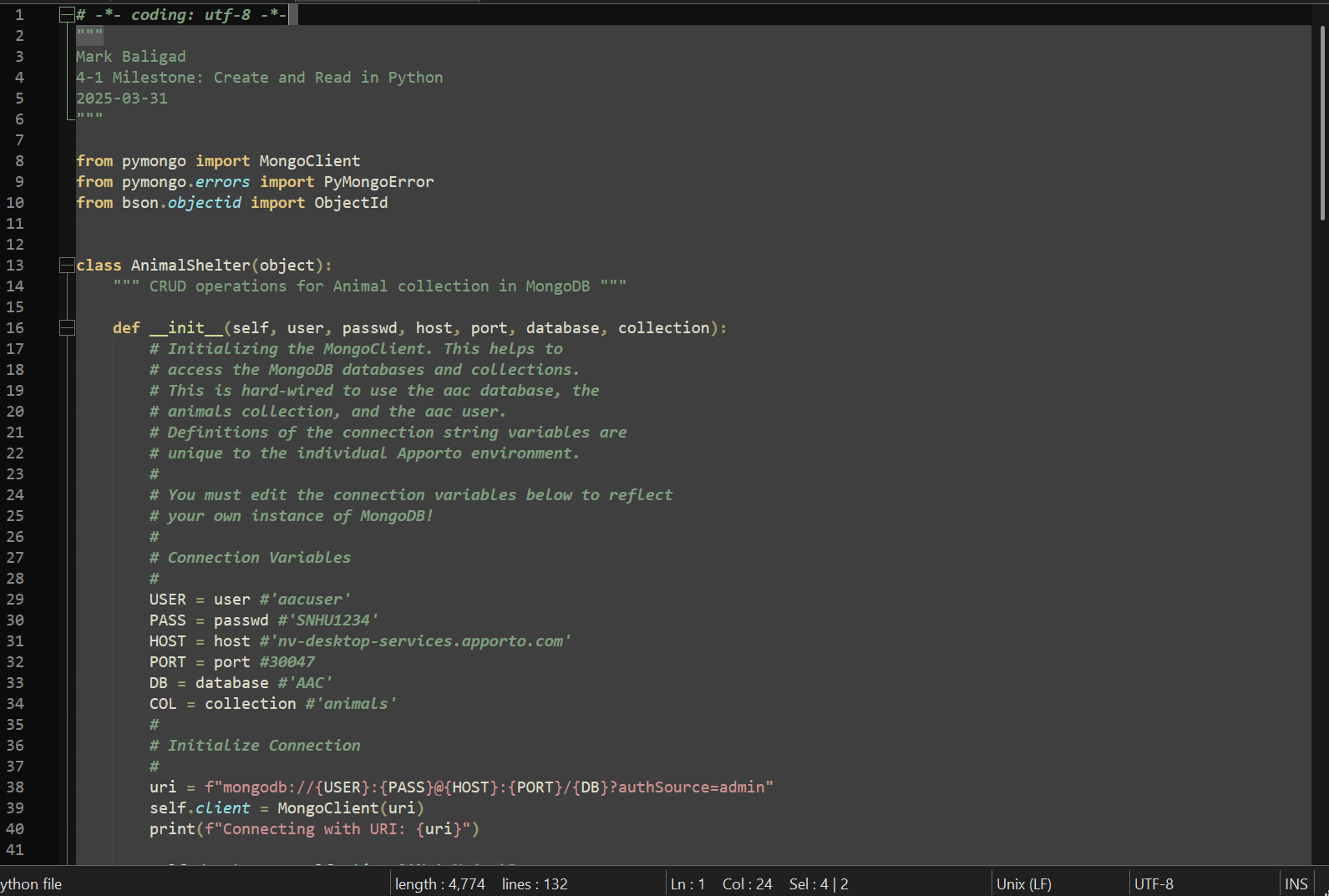
* [MongoDB](https://www.mongodb.com/try/download/community): MongoDB is a NoSQL, document-oriented database that stores data in a flexible, JSON-like BSON format for high scalability and performance.
  + Includes user authentication with varying degrees of access for the “admin” (full access) and “aacuser” (read/write access) credentials.
  + For this project, MongoDB facilitates storing documents for each Shelter Animal along with their characteristics. MongoDB facilitates authentication and provides an excellent user experience due to its scalable performance.
* [Python](https://www.python.org/):
  + The [Pymongo](https://pymongo.readthedocs.io/en/stable/index.html) module facilitates interaction with MongoDB to perform create, read, update, and delete functions.
  + The [Dash](https://dash.plotly.com/r) module is an open-source framework developed by Plotly for building interactive web applications for data visualization.
  + The [Pandas](https://pandas.pydata.org/) module provides us with tools for data manipulation and analysis.
  + For this project, Pymongo enables our CRUD module to access our Animal Shelter records. Pandas organizes the downloaded data for use by the Dash module, which displays the data in an interactive web application.

## Installation

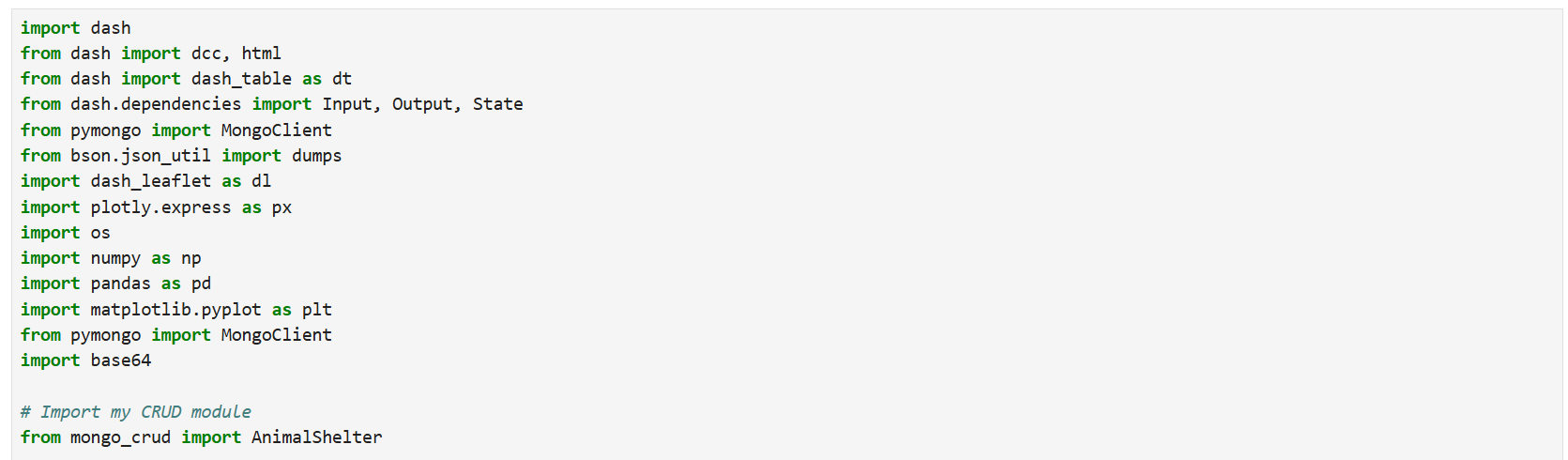
1. Install Python.
2. Install MongoDB.
3. Launch your Python IDE.
4. Start a new Python project and import the AnimalShelter class from the mongo\_crud file (see usage section for examples).
5. Follow the usage instructions below to build your interactive web application.

## Usage

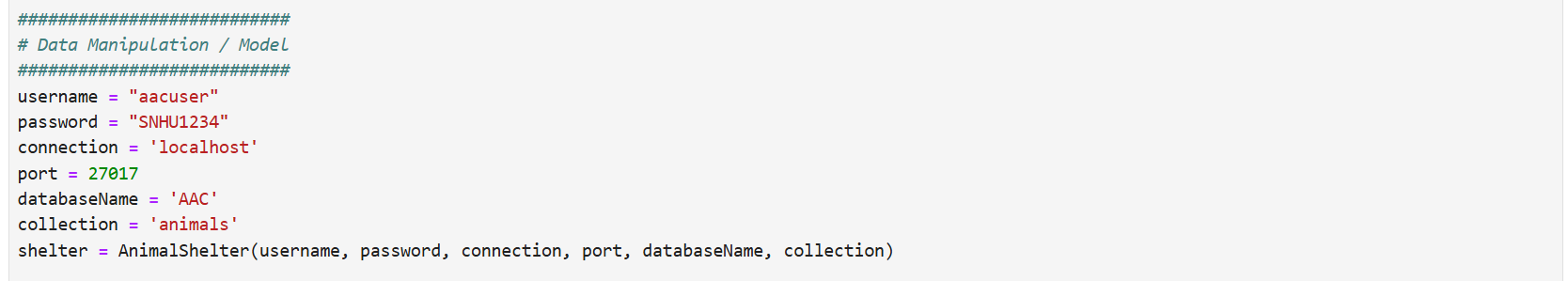
1. Store the mongo\_crud.py file in your project folder.

**

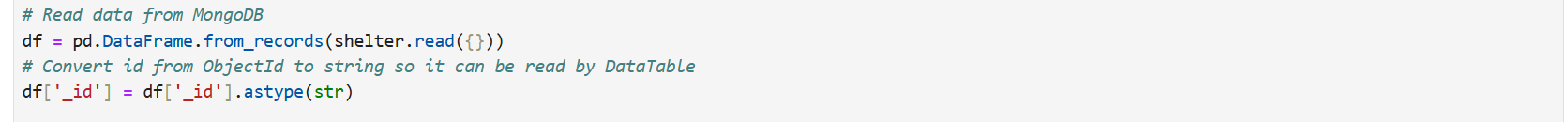
1. In a new Python file, import all the required Python modules.



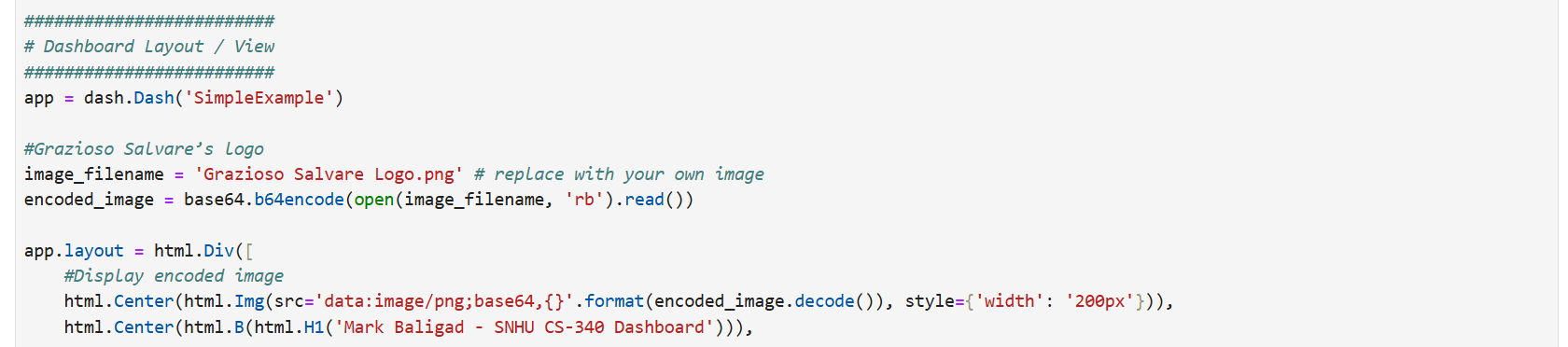
1. Create an instance of the AnimalShelter class and pass the MongoDB connection details as parameters.



1. Load data from MongoDB into a Pandas data frame.



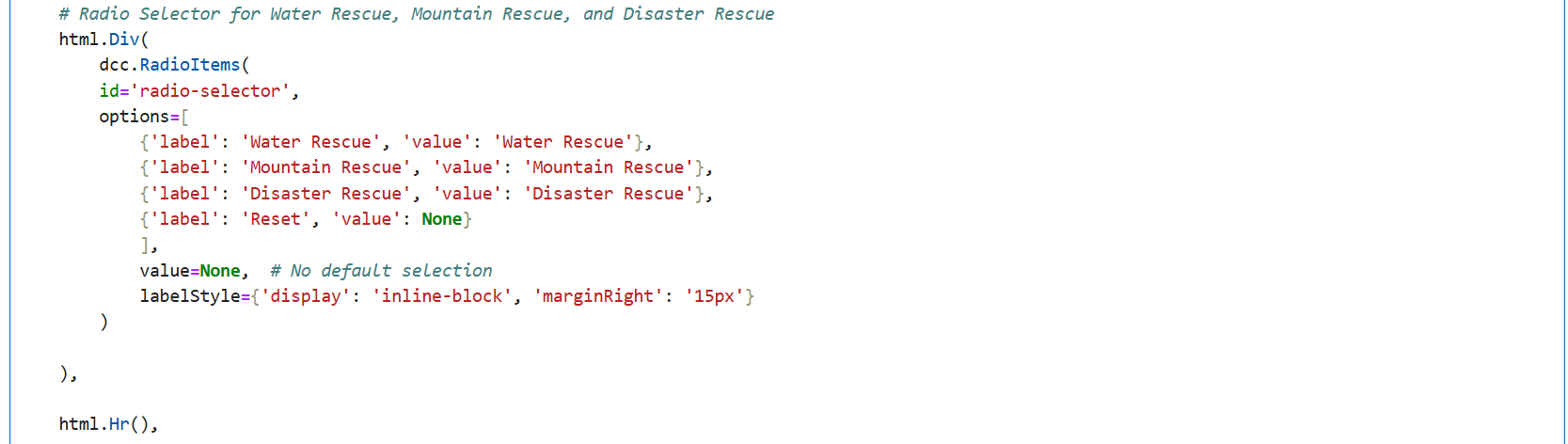
1. Begin creating the layout of the Dash web application.



1. Create a drop-down filter for animal\_type, breed, and color.



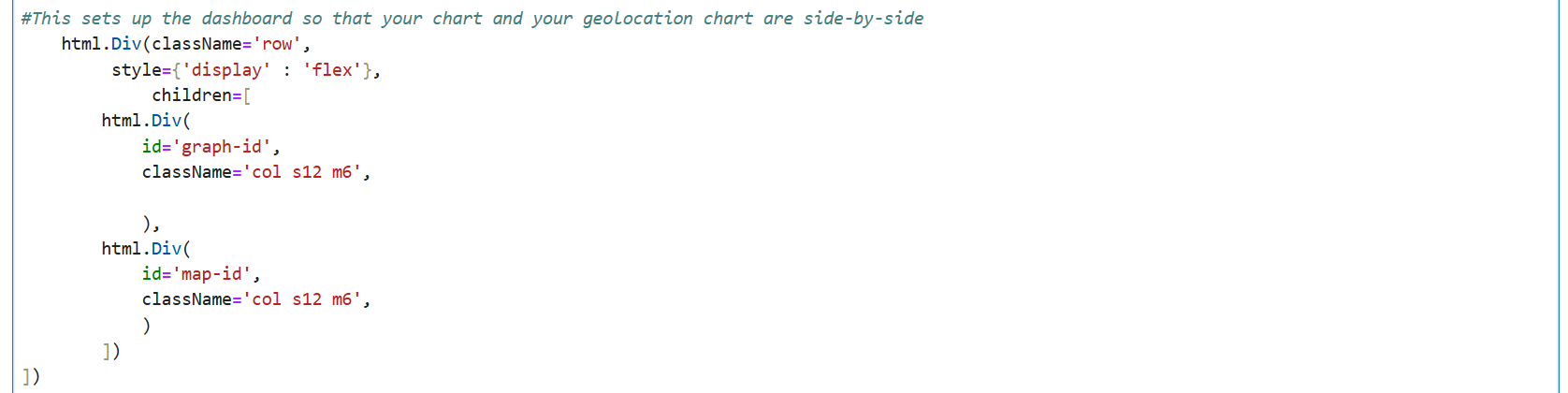
1. Create a radio selector for Water Rescue, Mountain Rescue, and Disaster Rescue.



1. Create a data table with a fixed height that the user can scroll through.



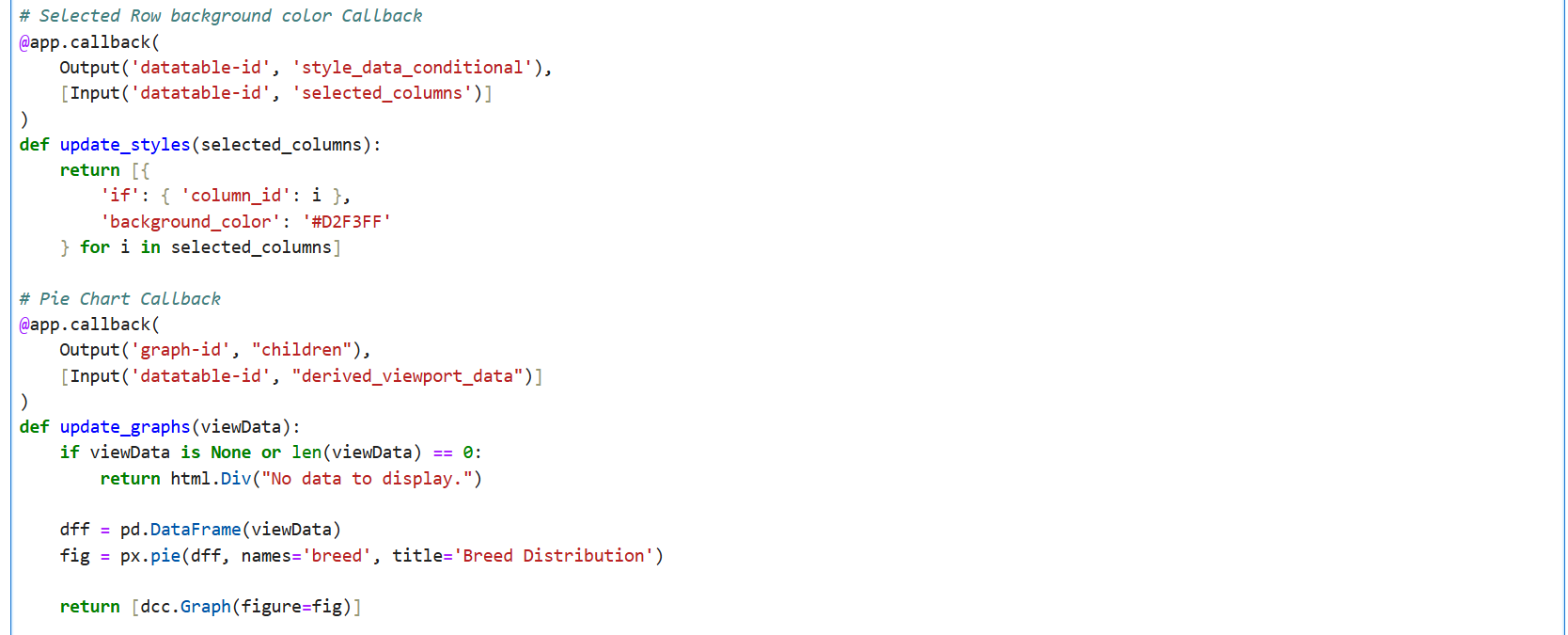
1. Create a pie chart and geolocation chart side-by-side.



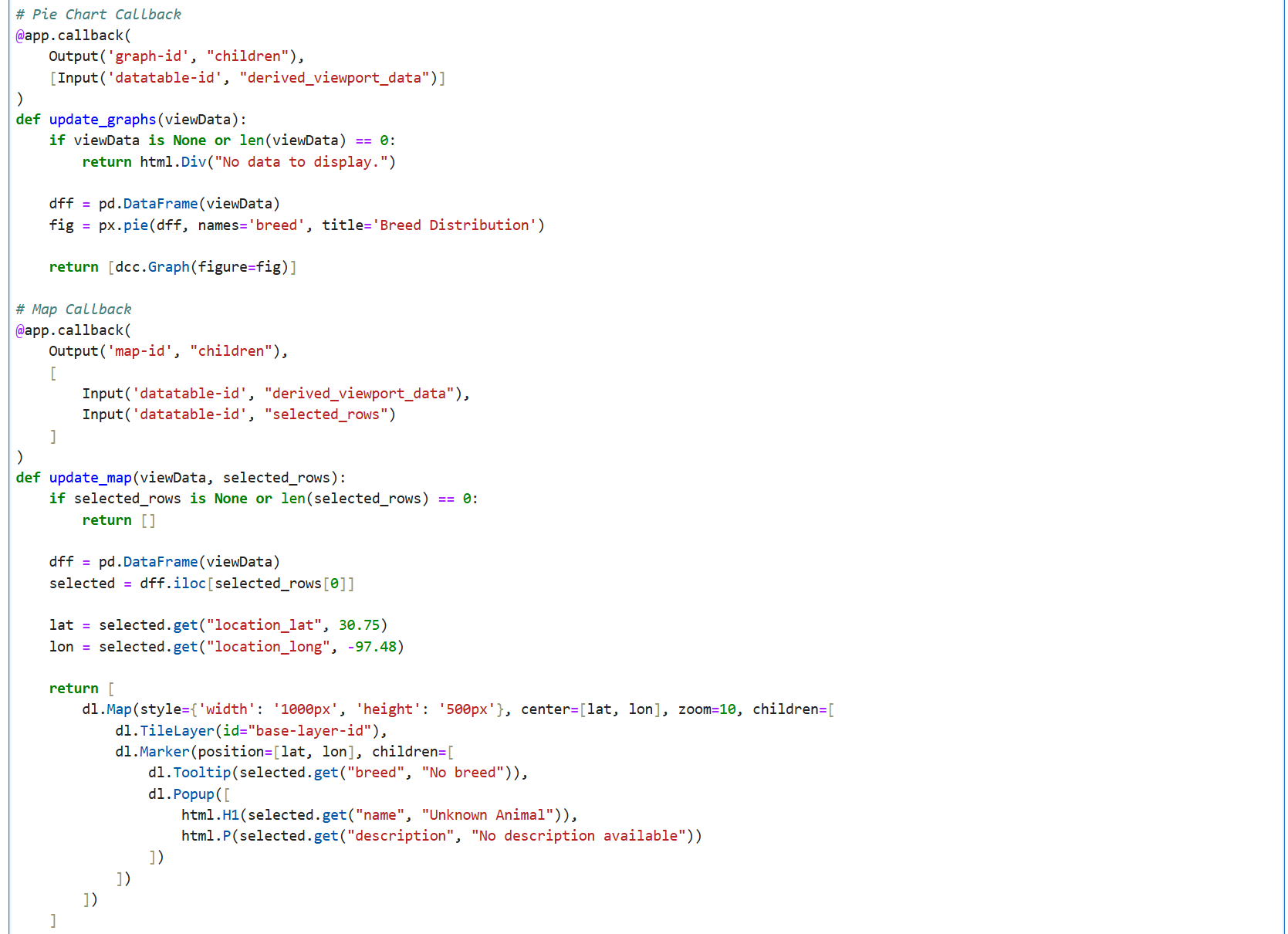
1. Define a callback function to update the data table using the drop-down and radio button options.



1. Define a callback function to update the background color of the data table cells when they are selected.



1. Define callback functions to update the pie chart and geolocation chart.



1. Run the app with the app.run() command.

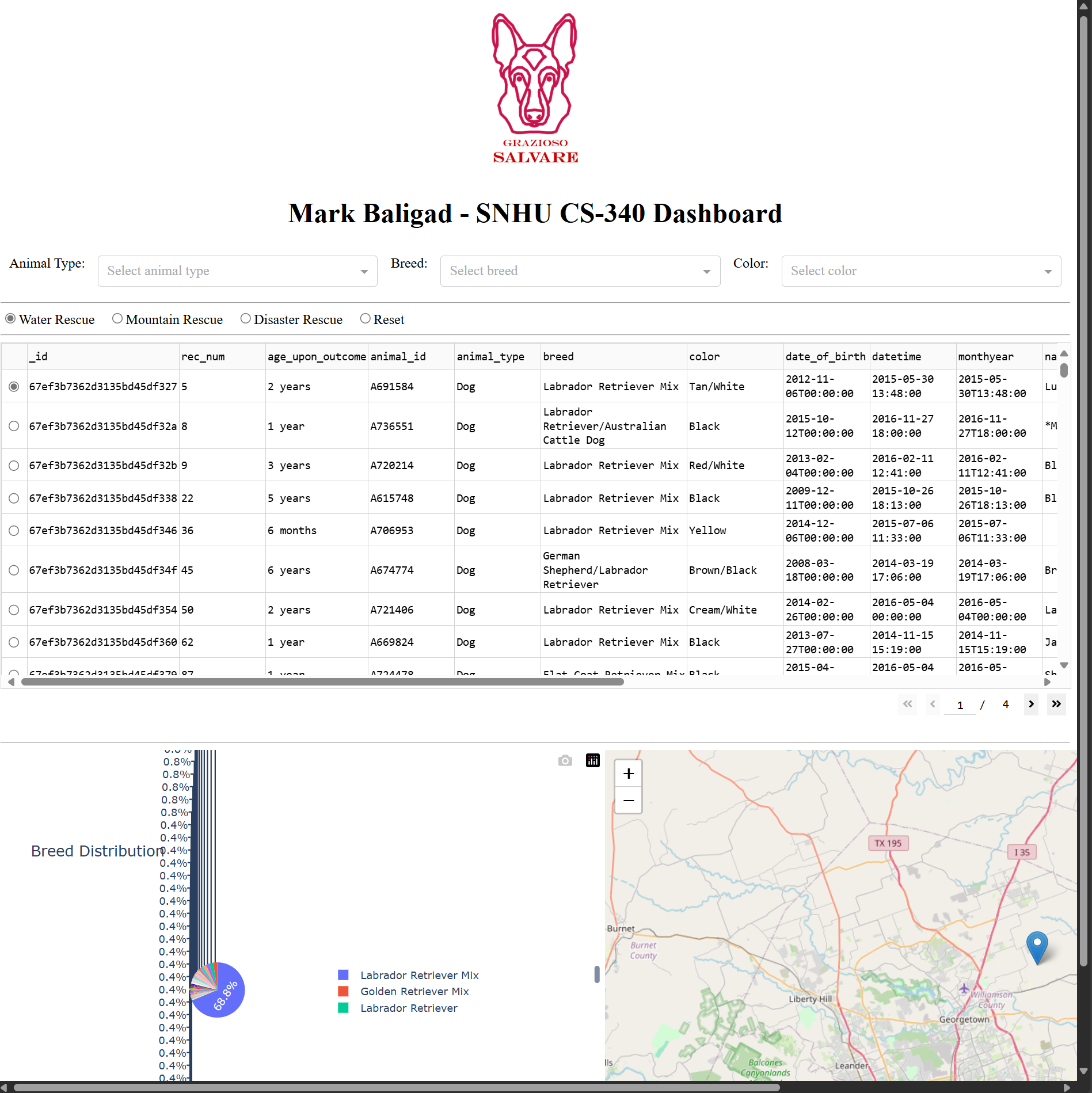


1. Open a browser and navigate to <http://127.0.0.1:8050/> to view your web application.

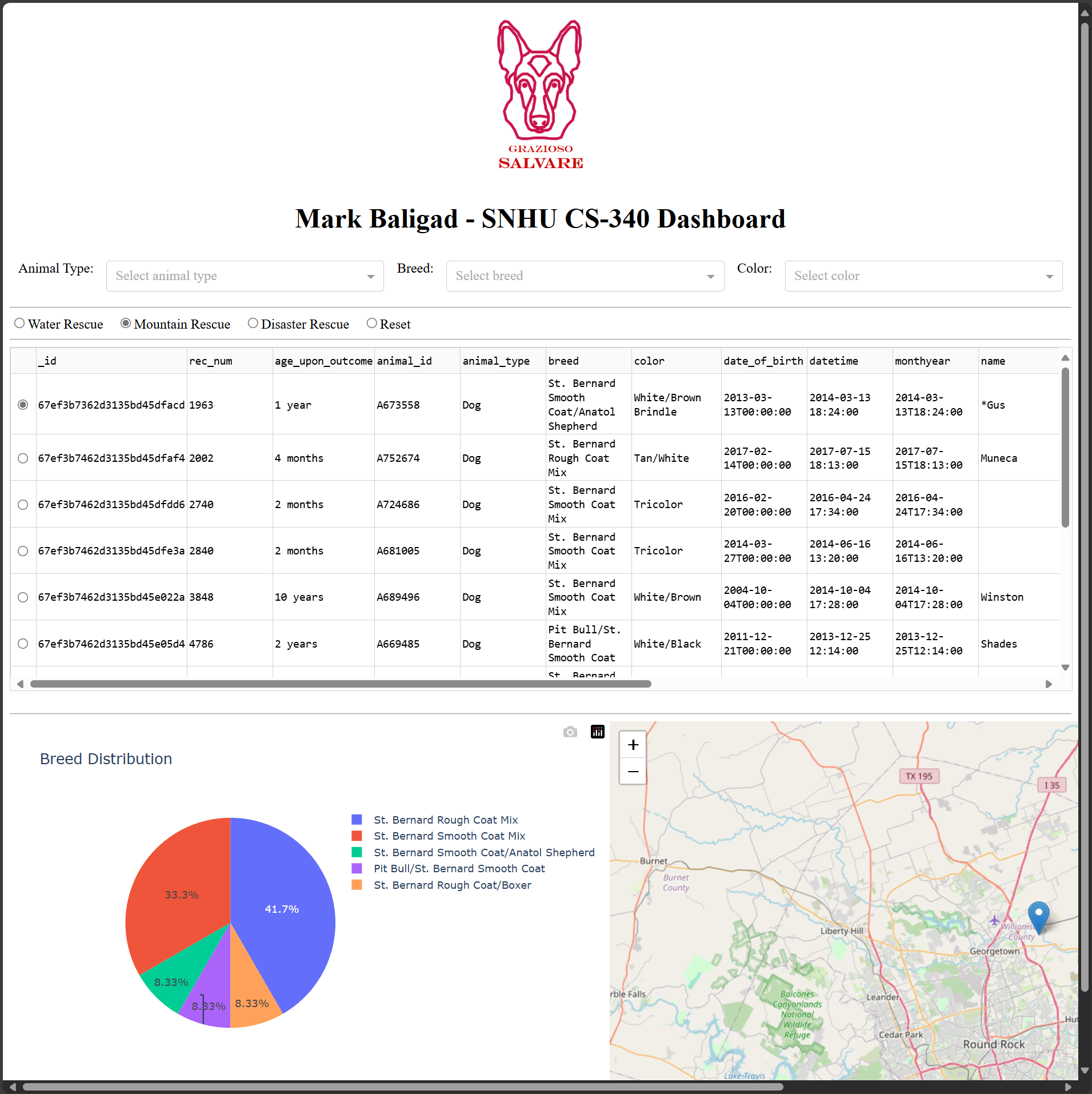


**Sample Screenshots**

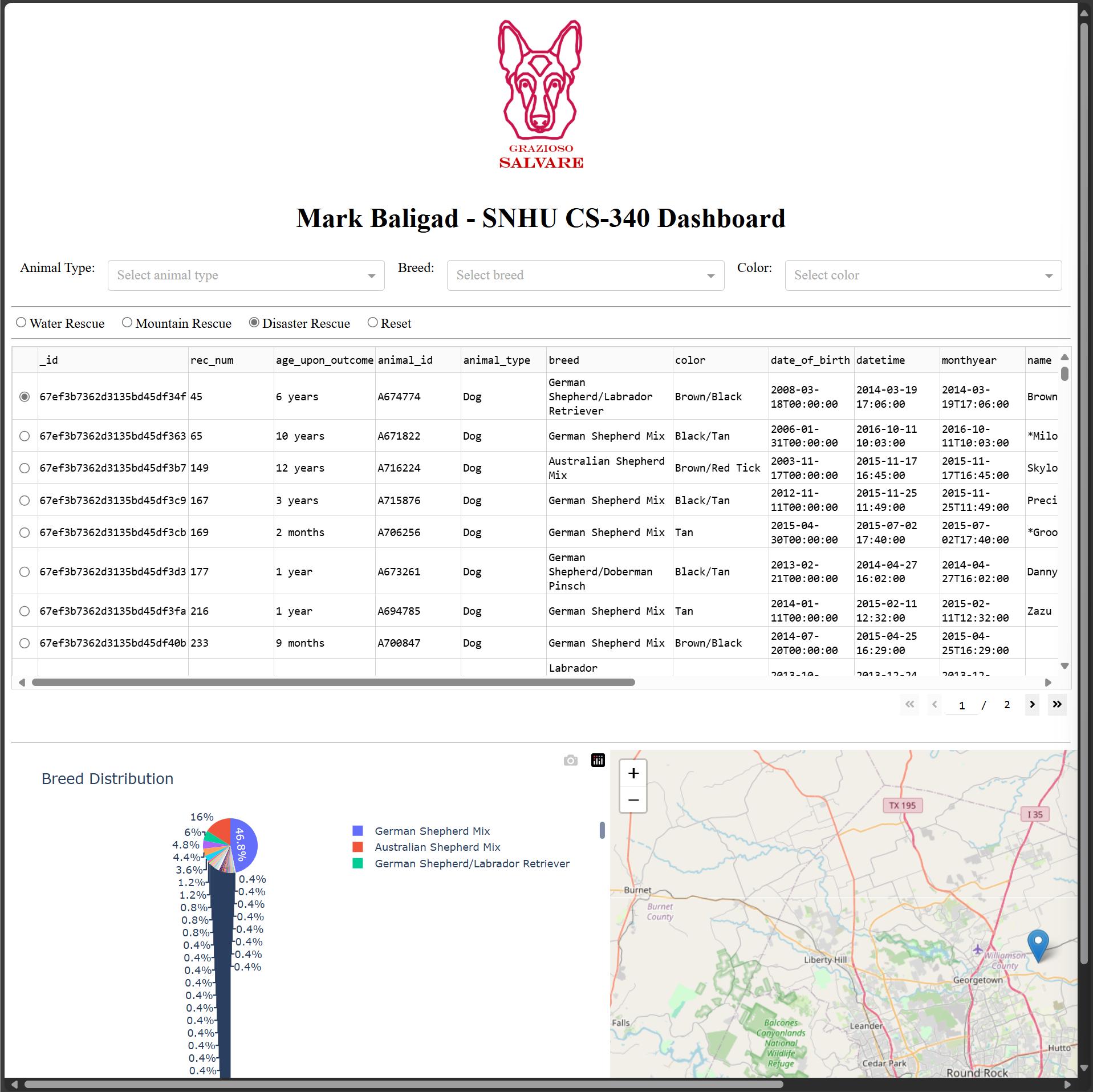
Example Filter: Water Rescue



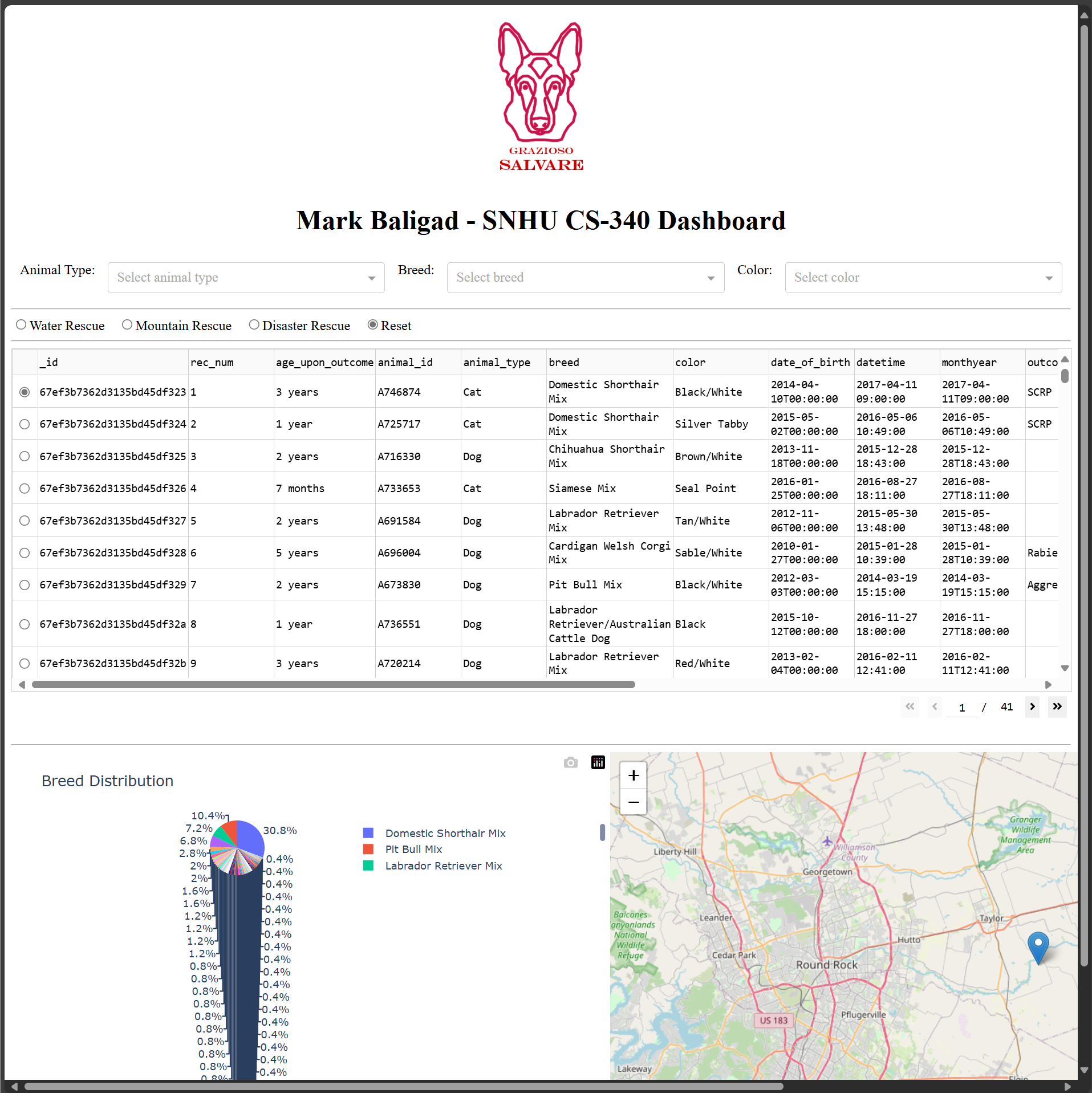
Example Filter: Mountain or Wilderness Rescue



Filter: Disaster or Individual Tracking



Filter: Reset



## Challenges

* A significant number of tutorials posted online reference outdated versions of the Dash module, such as the now-obsolete Jupyter Dash module. Be sure to reference the most recent material directly from <https://dash.plotly.com/r>
* I learned that you cannot have multiple callback functions affecting the same output; they will conflict. Include all your inputs and filters under the same callback function to solve this. For example, see the callback function for the drop-down and radio button options above.

## Contact

Your name: Mark Baligad <mark.baligad@snhu.edu>