# CS 340 Project Two README

## About the Project

This project was created for the company Grazioso Salvare to search for dogs who are good candidates for search-and-rescue training. To achieve this, we will go through the Austin Animal Center Outcomes data set found in MongoDB and create a CRUD module in python/Jupyter Notebook to access this data. The client web application dashboard will be created in Python to give users access to the Austin Animal Center database. Through the dashboard, they can filter through the animals to find the perfect candidates for training.

## Getting Started

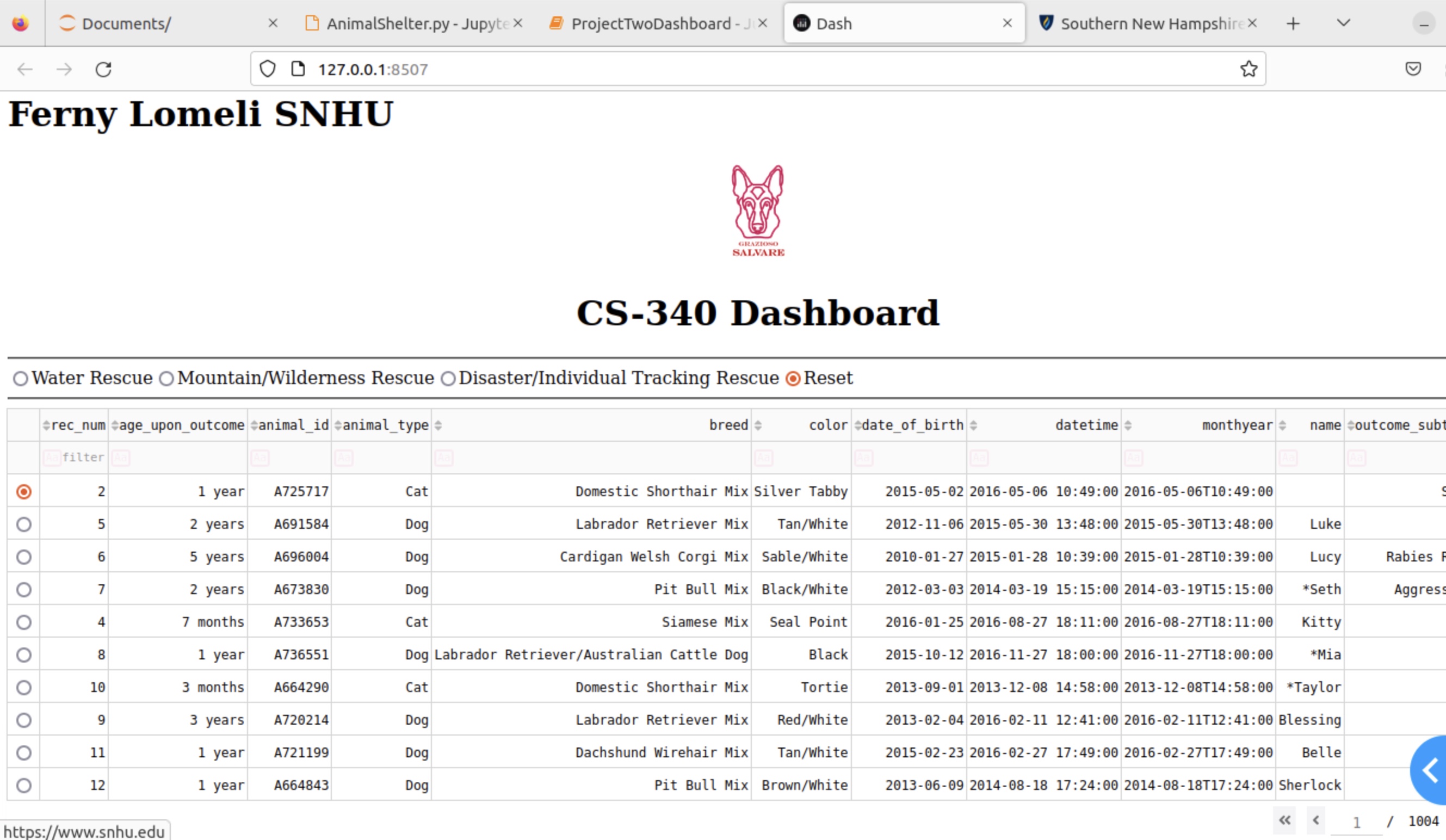
You must complete this series of steps to set up the project yourself. To begin, you must have access to the Linux shell and upload the Austin Animal Center (AAC) Outcomes data set into a new database called AAC in MongoDB. After creating the AAC database, you will open the Mongo shell and create a simple and compound index to enhance the performance of the queries. Next, you should set up user authentication by creating admin and user accounts with both having access to the AAC database. The create, read, update, and delete functions are then established in Jupyter Notebook to create, read, update, and delete documents in the database. These functions can be obtained using the AnimalShelter.py file. From there, you can use the Python ProjectTwoDashboard.ipynb file which was used to access the AAC database through the AnimalShelter.py file and create dashboard widgets from the data for the client web application. The Model View Controller is the software design pattern used for this multi-tier application. The model is accessed/contained in MongoDB, the views are the dashboard widgets obtained by a RESTful API, and the controller uses the AnimalShelter.py (CRUD Python Module) to use the queries and interact between the components.

## Installation

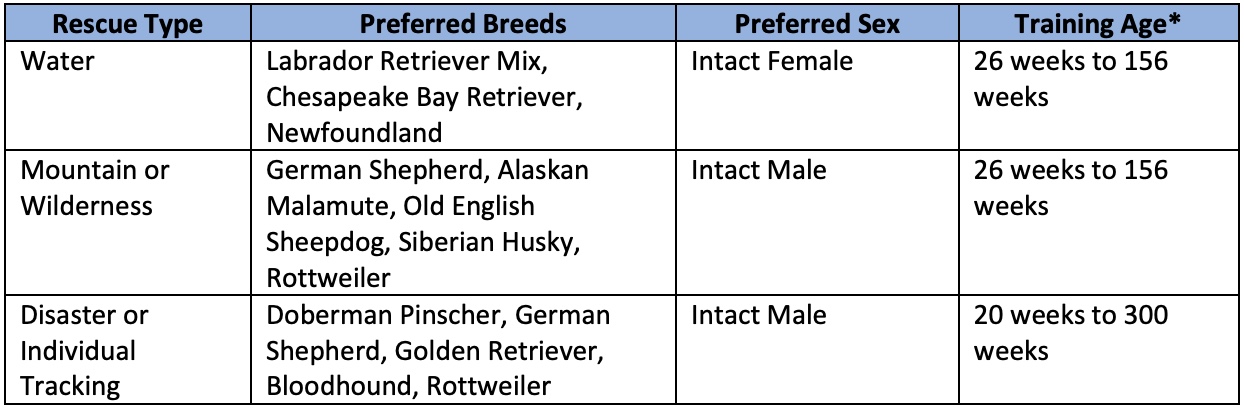
There are a few tools required to use this software. The first tool required is the Linux terminal GNOME. This will provide access to the Linux shell and Mongo shell to create the database and user authentication. MongoDB can be installed here https://www.mongodb.com/try/download/community. Jupyter Notebook can be used to create the PY file and the IPYNB file. You can find the installation for it using this link https://github.com/jupyter/notebook. You can find the installation for Python here https://www.python.org/downloads/. Download Dash here https://pypi.org/project/dash/. Installing Pandas can be found here https://pandas.pydata.org/pandas-docs/stable/getting\_started/install.html.

## Usage

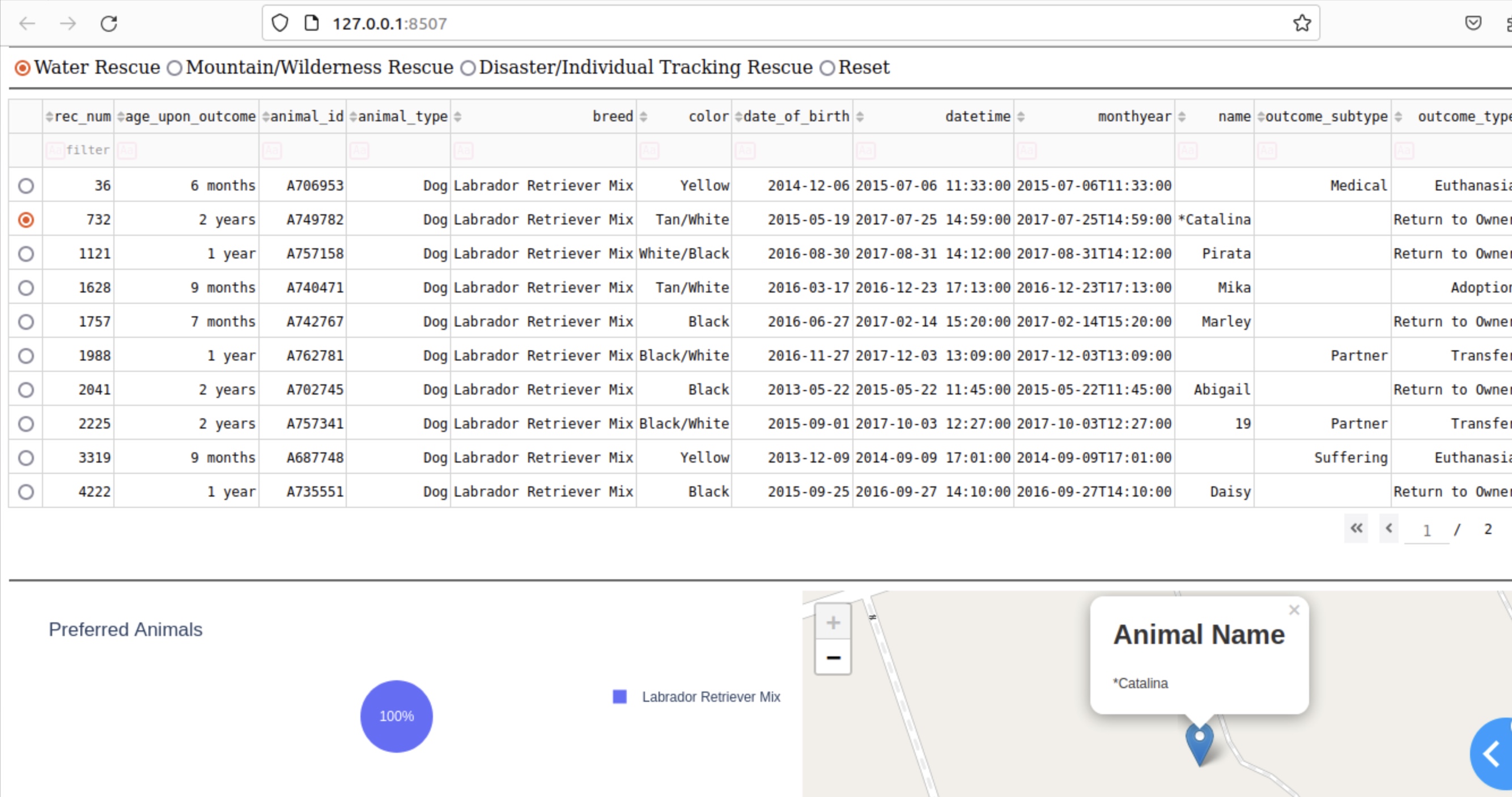
The client dashboard, that has access to the Austin Animal Center Outcomes database, includes the data table displaying the data, radio items for filtering options for the data table, a pie chart relaying the data table information, and a geolocation chart for the selected animal. The image below displays the dashboard with the Grazioso Salvare logo that has a link to the SNHU homepage.

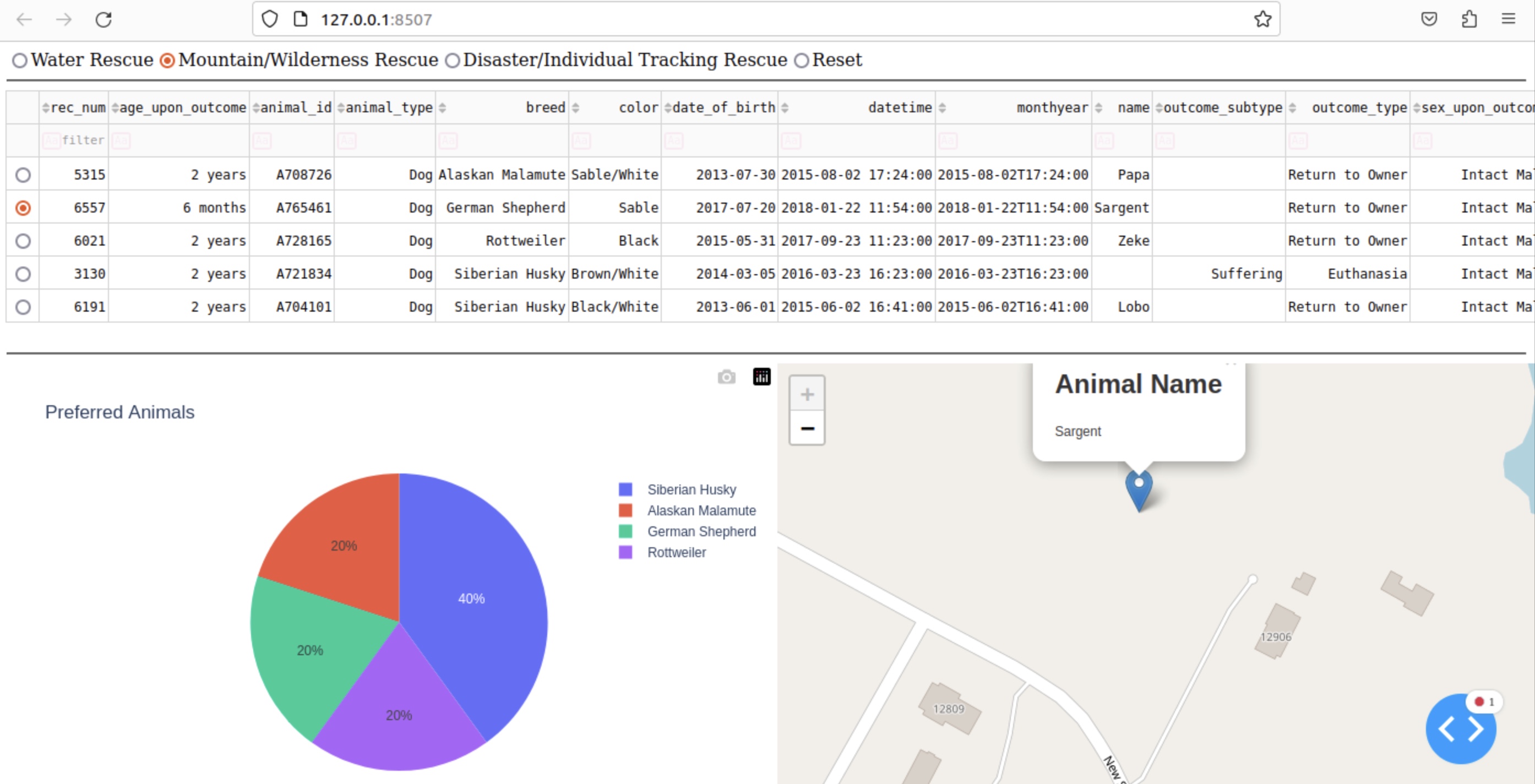


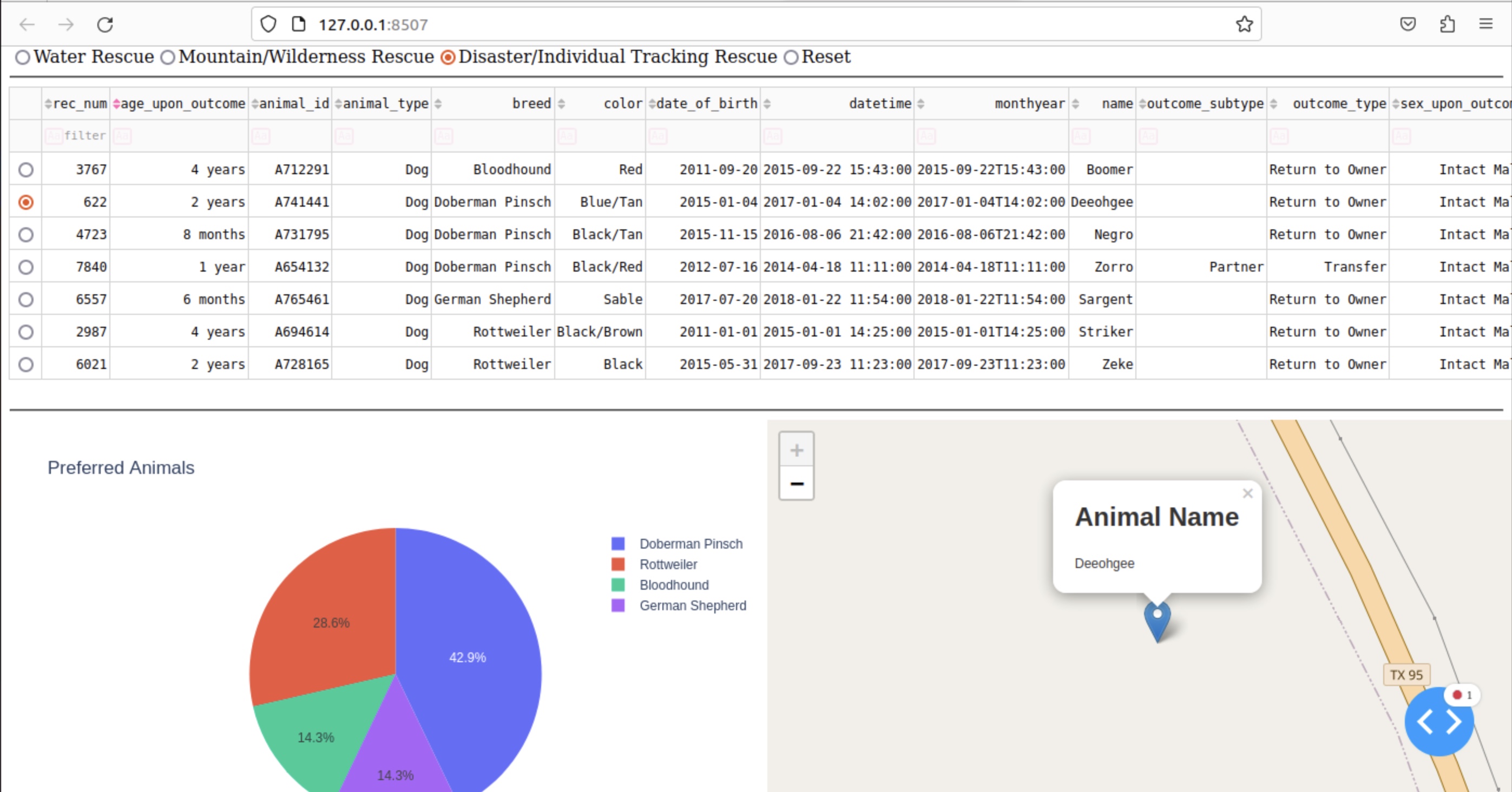
The image below displays the filtering options required by Grazioso Salvare for the rescue type of dog they are seeking.

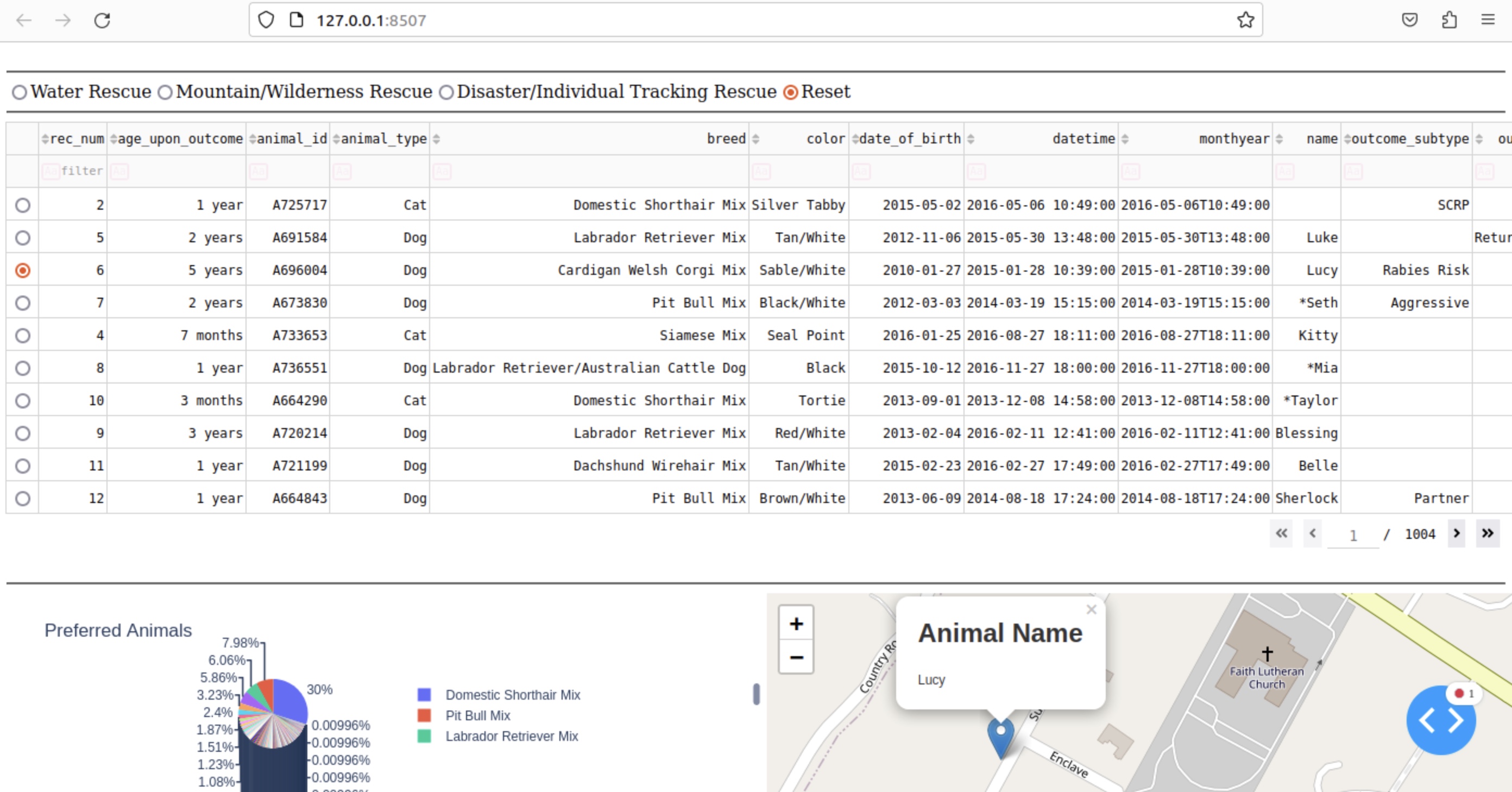


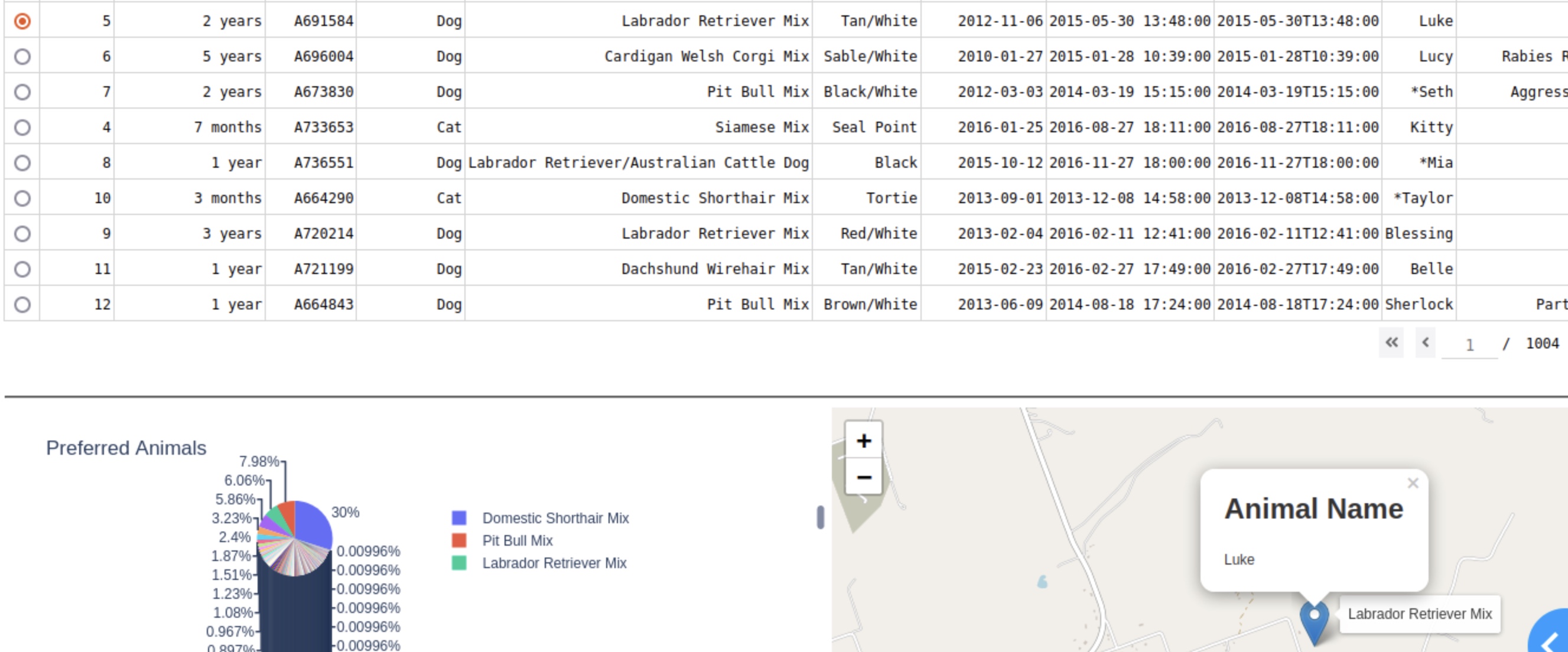
The image below displays the eligible dogs found for Water rescue training by selecting the Water Rescue filtering radio item option. The pie chart and geolocation chart are updated according to the filter settings.

The image below displays the eligible dogs found for Mountain or Wilderness rescue training by selecting the Mountain/Wilderness Rescue filtering radio item option. The pie chart and geolocation chart are updated according to the filter settings.

The image below displays the eligible dogs found for Disaster or Individual Tracking rescue training by selecting the Disaster/Individual Tracking Rescue filtering radio item option. The pie chart and geolocation chart are updated according to the filter settings.

The image below displays the Reset filtering option being selected and resetting the widgets to their original, unfiltered state.

The image below shows how the geolocation chart displays the animal's name and breed when hovering over the map marker.



**Challenges**

A few challenges were encountered while trying to complete this project. Some of the animal names such as Doberman Pinscher were not written out completely so the filtering option of “Doberman Pinscher” would not display the animal. Instead, the filtering option of “Doberman Pinsch” would have to be used. Next, in the AnimalShelter.py document, for the Read function the data needed an {“\_id”:False} argument to be passed for the radio item filtering options to work (data = self.database.animals.find(queryData, {"\_id":False})).

## Contact

Your name: Fernando Lomeli

Email: Fernando.Lomeli@snhu.edu