# CS 340 – Grazioso Salvare - Animal Shelter Dashboard

## About the Project

*Grazioso Salvare is seeking a software application that can work with existing data from the animal shelters to identify and categorize available dogs for rescue operations. The data for these animals has been provided and stored in the “AAC” (Austin Animal Center) database. That data was then configured to display in the custom dashboard using the client requested filters.*

## Motivation

*This project is dedicated to Professor Kellogg, without his diligent instruction and helpful feedback, this project would have probably never come to fruition. The main motivation behind this project was the knowledge that was obtained while working through each step and sharpening my Jupyter Notebook, Dash, Python and MongoDB skills.*

## Tools used to achieve functionality

*All the data is stored and configured within a NoSQL database called MongoDB. The data within the AAC dataset while structured was very efficiently indexed, utilized, and queried using MongoDB. The biggest benefit of using Mongo however was the interfacing with Python, the language used to build the dashboard.*

*The PyMongo Driver made the connection between the Bson’s provided by Mongo and the HTTP displays encoded in Python. I used Jupyter Notebook to write the majority of the code and Dash Framework to provide all the views and controllers. The Dash framework is an HTML connection that can be used in Python to display web applications and interfaces.*

## Steps taken to complete the project

*The steps taken to complete this project were as follows. Install the latest versions of MongoDB, Python, and PyMongo. Acquire the dataset and launch an instance of Mongo. Import the dataset into the database and create an authentication for users to access the data. Create a workable index on the data to allow for fast and efficient querying of the dataset. Once the dataset is ready, a Python Class has to be written to act as an API to query the database. The basic elements for this class are within the acronym CRUD (Create, Read, Update, Delete). CRUD is the basic functionality required to access the database ant utilize it in front end web application. The python class then must be tested using different calls to make sure it performs flawlessly before moving on to the next part.*

*The web interface is then written using a program called Jupyter notebook using the Python Object-Oriented Programming. All the elements required for this interface need to be imported first, following that the database has to accessed and authenticated, initiating the dataset and the python classes. The layout is then designed using Dash elements and callbacks are used as controllers for those elements.*

## Challenges

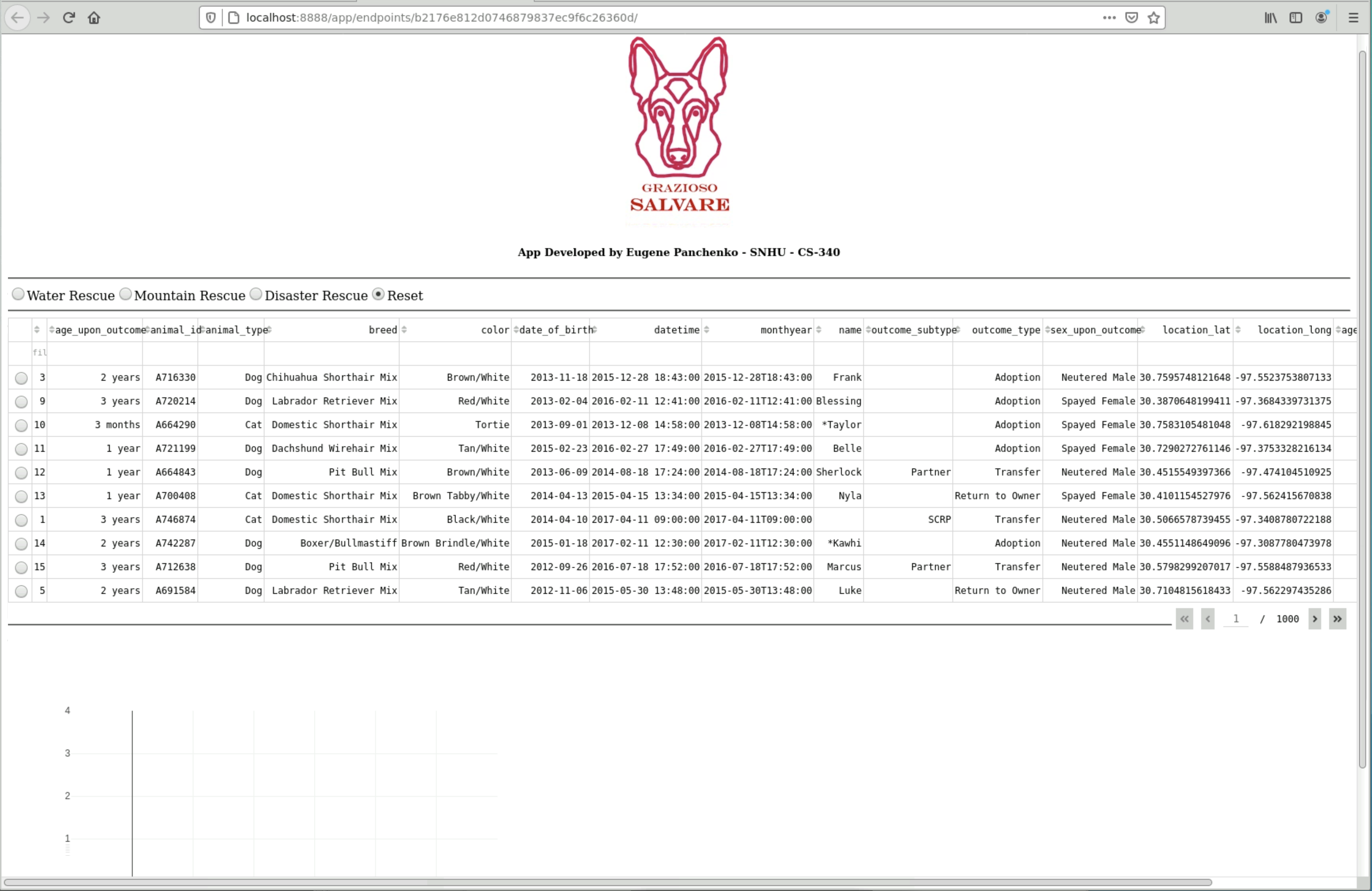
*Oh where to begin! There were many! Joking aside the main challenges I faced were centered around learning a new language. The behavior, syntax and commands of MongoDB were very straightforward, and I quite enjoyed working with the language once I grew more comfortable with it. I was doing well until it came time to write the Python classes. I had a tough time wrapping my mind around what the calls were requesting since it wasn’t immediately clear if it was pulling a record, document, list or dictionary. While testing that class the Jupyter notebook wasn’t very helpful because if something was amiss the code simply wrote “loading…”, displayed a blank output or best-case scenario gave me an error message.*

*The majority of my challenges with this final project revolved around the interface specifically the data and formatting of the Map and Pie charts. When making the dashboard layout I found it very difficult to build a layout especially when there was no rhyme or reason as to why things failed. My usual outcomes were either a blank output screen or a never-ending “Loading…” screen. It wasn’t until I discovered that the terminal, I used to call the jupyter notebook also provided error logs that things really started to come together. After that discovery, I was able to work through all my issues and put together a polished efficient dashboard that I am proud to submit.*

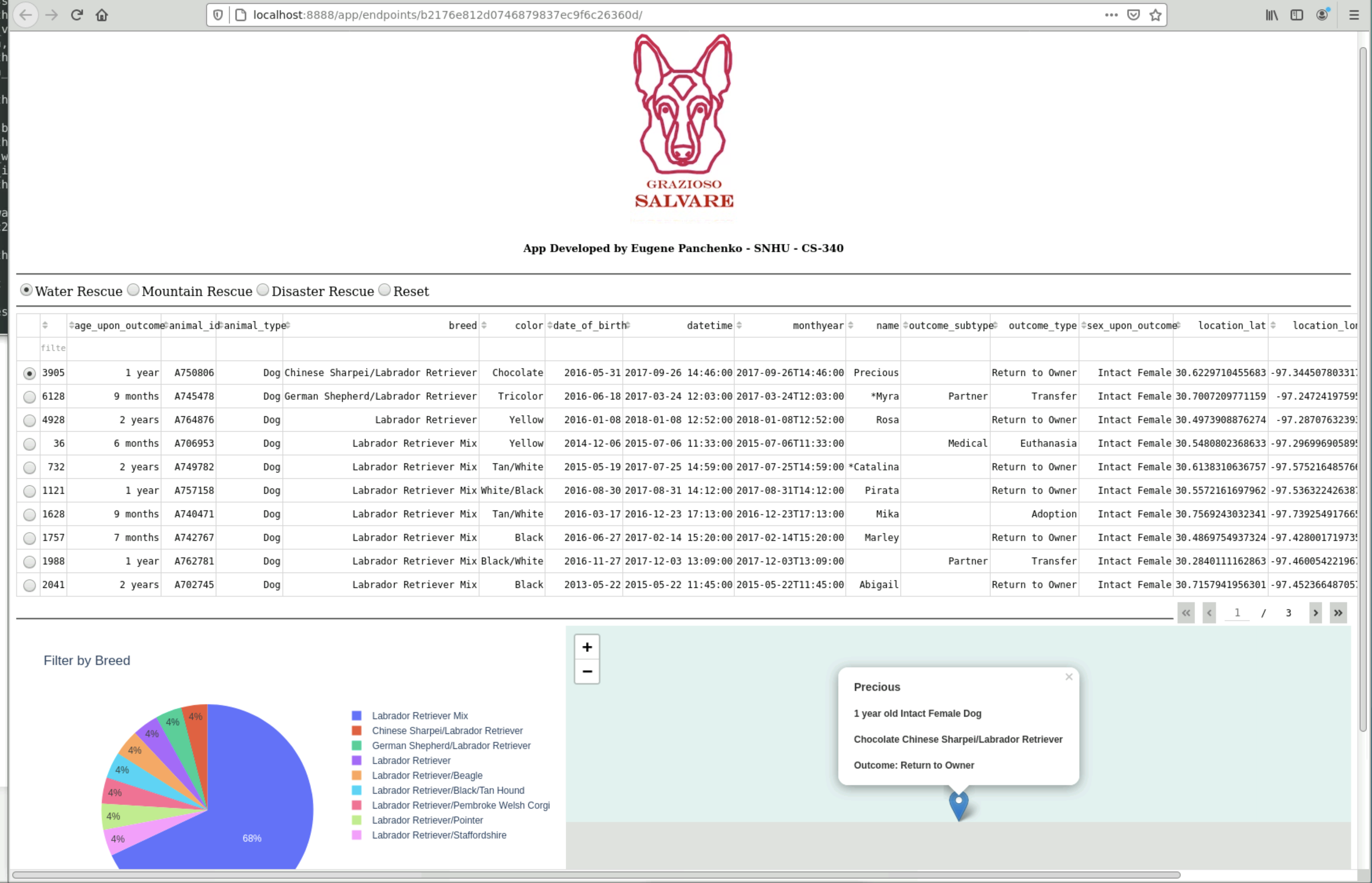
## Usage

### Screenshots

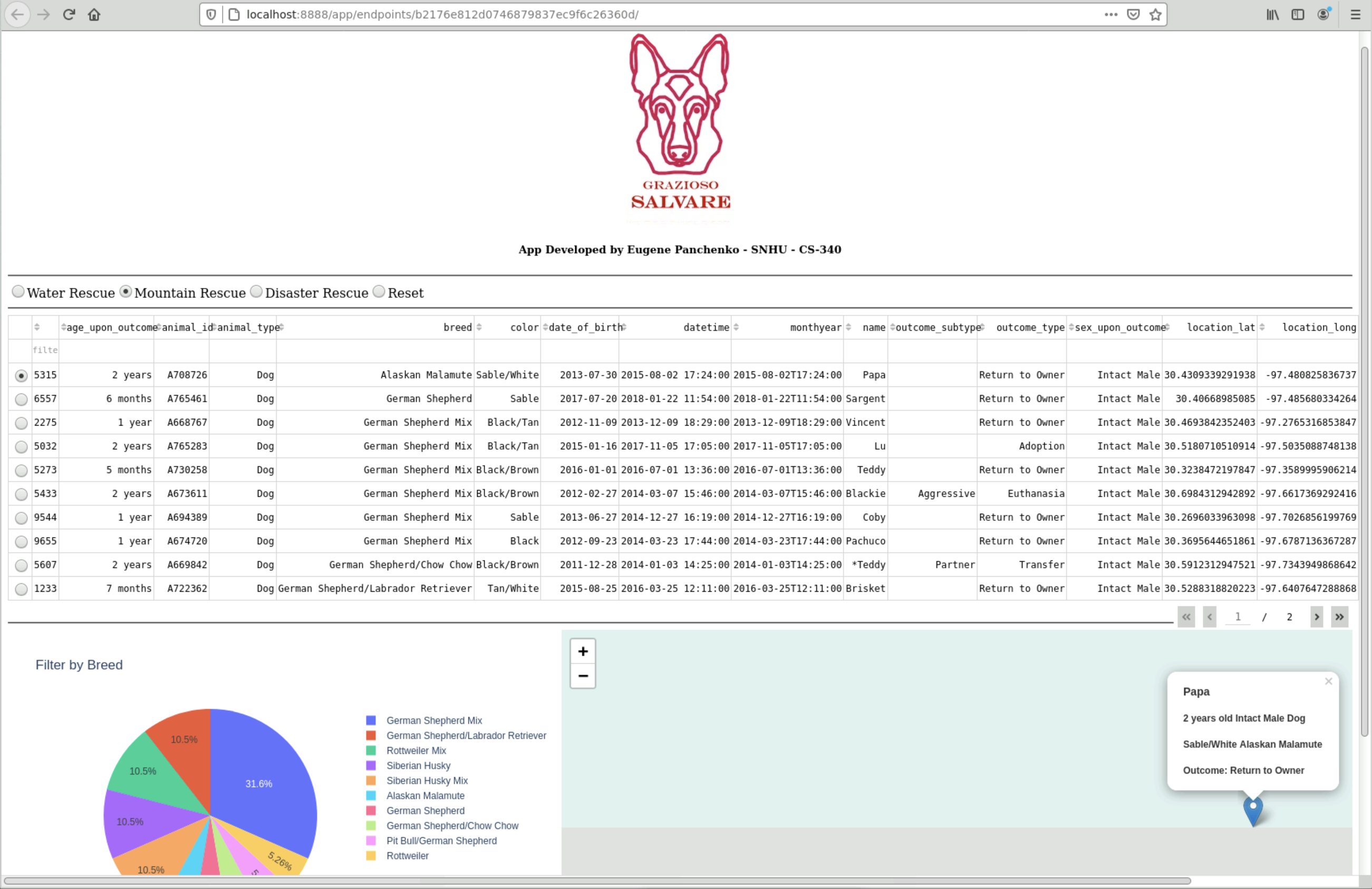
Starting state



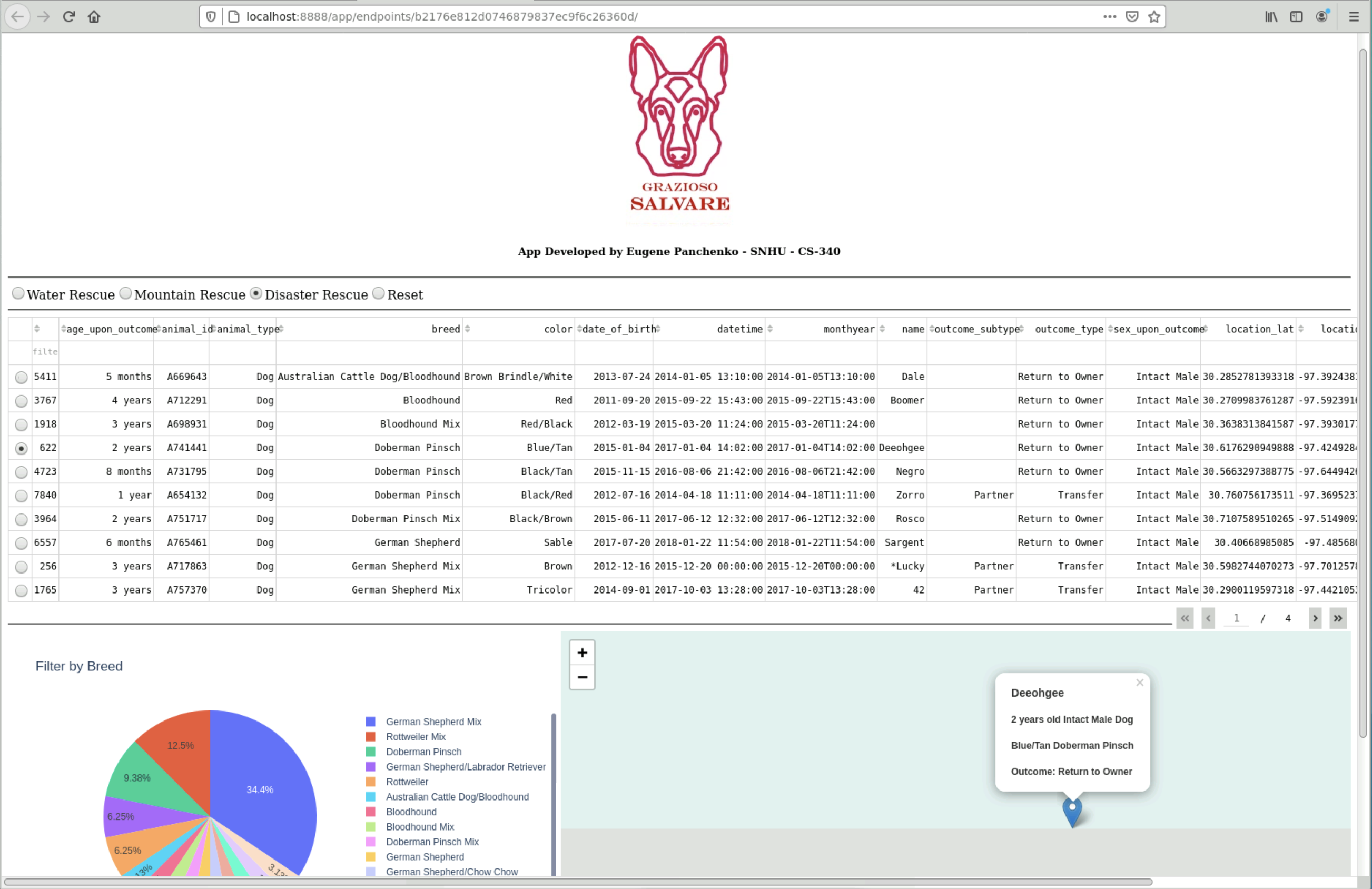
Water Rescue



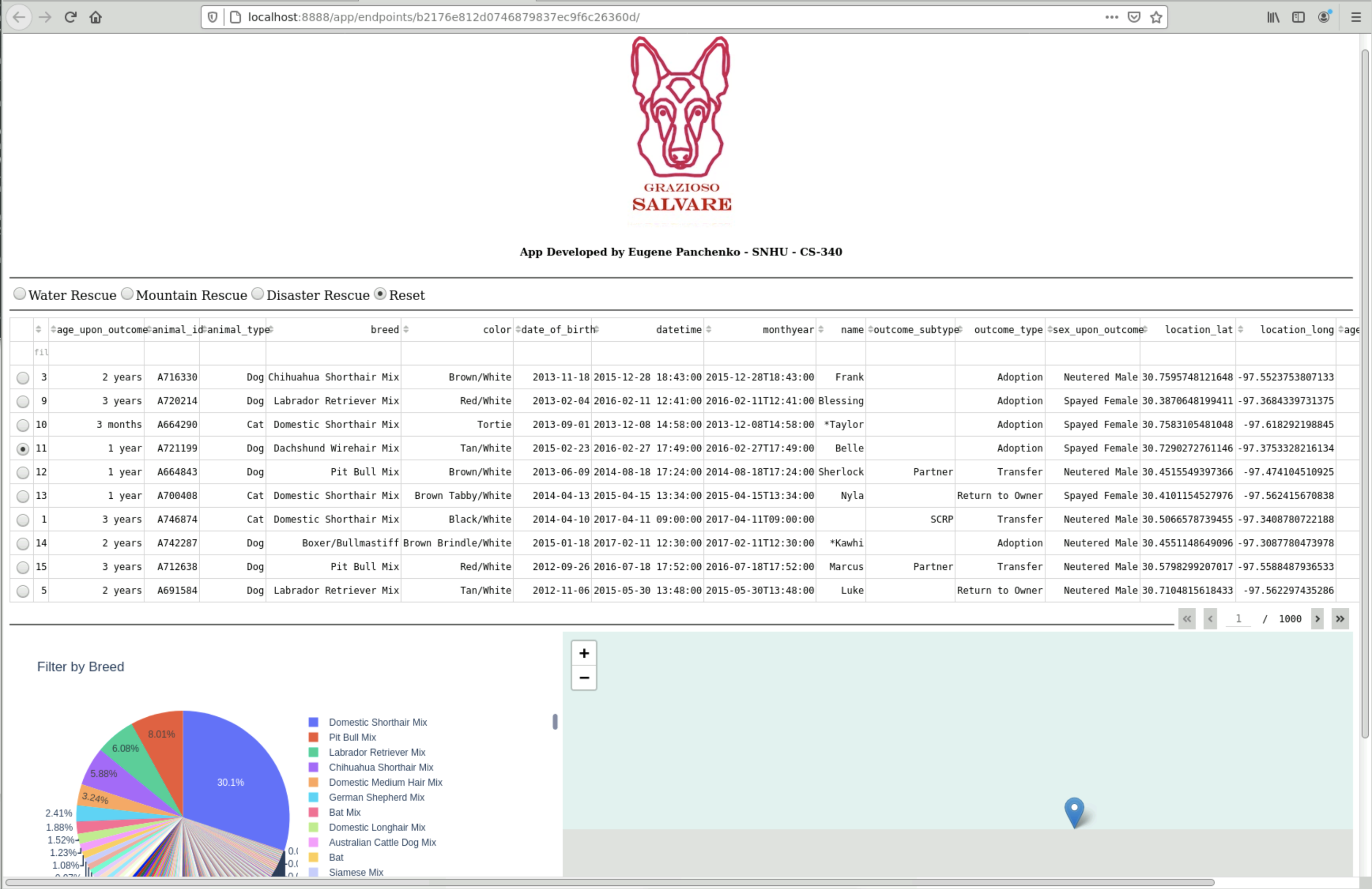
Mountain or Wilderness Rescue



Disaster or Individual Tracking



Reset (returns all widgets to their original, unfiltered state)



## Contact

Eugene Panchenko

Eugene.Panchenko@SNHU.edu