# CS 340 README Project: Grazioso Salvare Search and Rescue Training.

About the project: Grazioso Salvare Search and Rescue Training.

Grazioso Salvare helps in identifying animals for international search-and-rescue training to aid in rescuing animals and people. The project finds good candidates by searching the Austin Animal Center (AAC) database for dogs that fit the client’s needs for where the dog will be searching. Upon searching for the dogs that match, the program will identify those, who are available.

Usage:

The project utilizes the C.R.U.D. application made in Python to read through the AAC database and creates a UI that gives the user ease of access to the data that is housed in AAC. The UI is comprised of a dashboard with a database table, two widgets that provide data on the filtered animals, and a dropdown to filter the types of animals to be used depending on conditions.

Tools:

* Python – version (3.10+ due to match cases), --modules(Pymongo, Dash, JupyterDash, dash\_leaflet, plotly.express, pandas) 🡪 This was used for the C.R.U.D. application
* MongoDB 🡪 Was used for the cloud server that housed the AAC database.
* Jupyter Notebook 🡪 Was used to create the UI of the application.
* Access to database(user/admin)

How to set up the database:

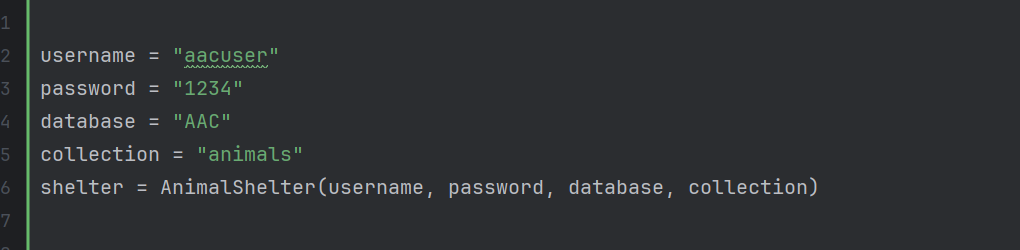
The following screenshots show uploading the \*.csv file to the MongoDB cloud and making a user for said database:

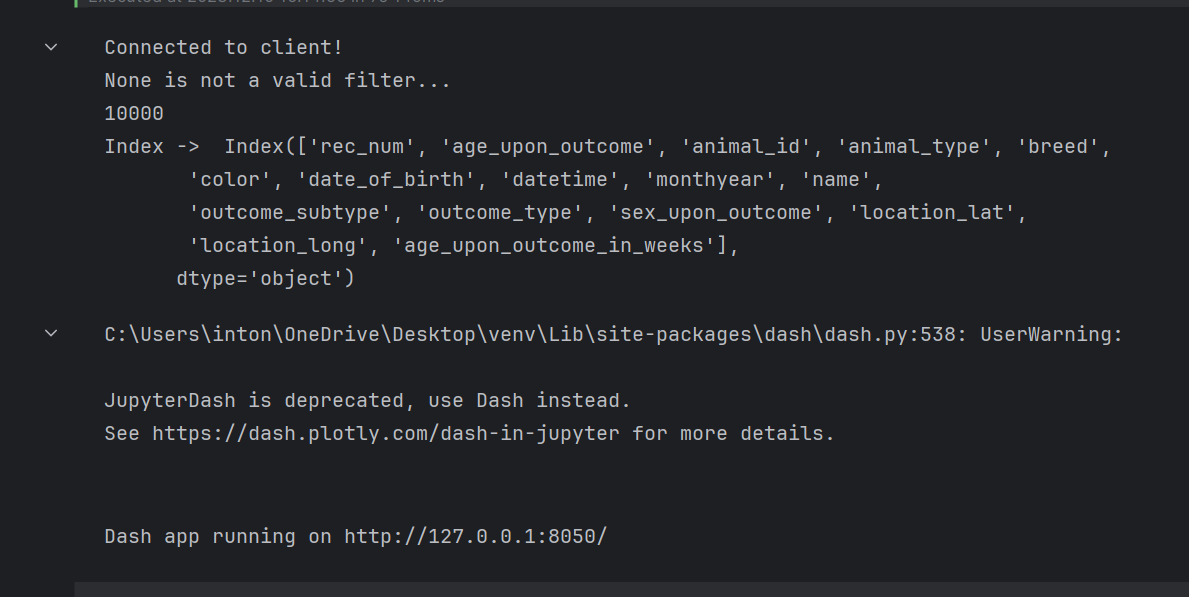


A screenshot of a computer screen

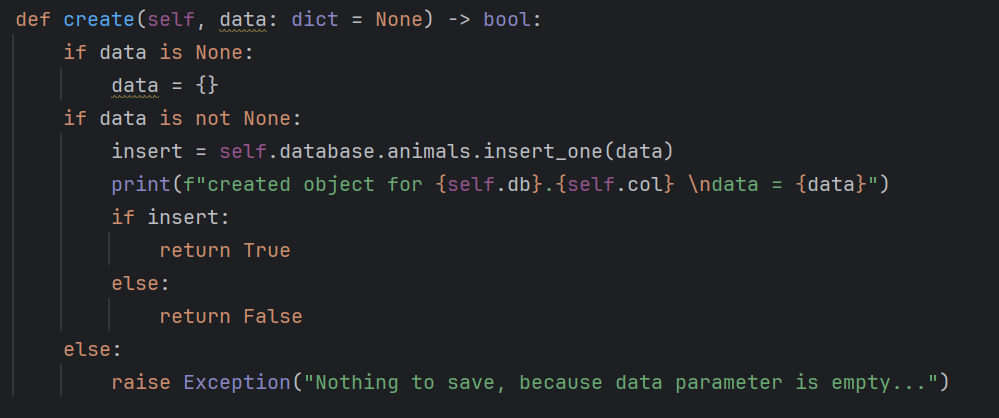
Description automatically generated

To see if you have access to the database you can open up a jupyter notebook and make a new file with importing the animal\_shelter.py script and upon creating the object, you will be greeted with “Connected to Client” is successful \*\*screenshot following\*\*:

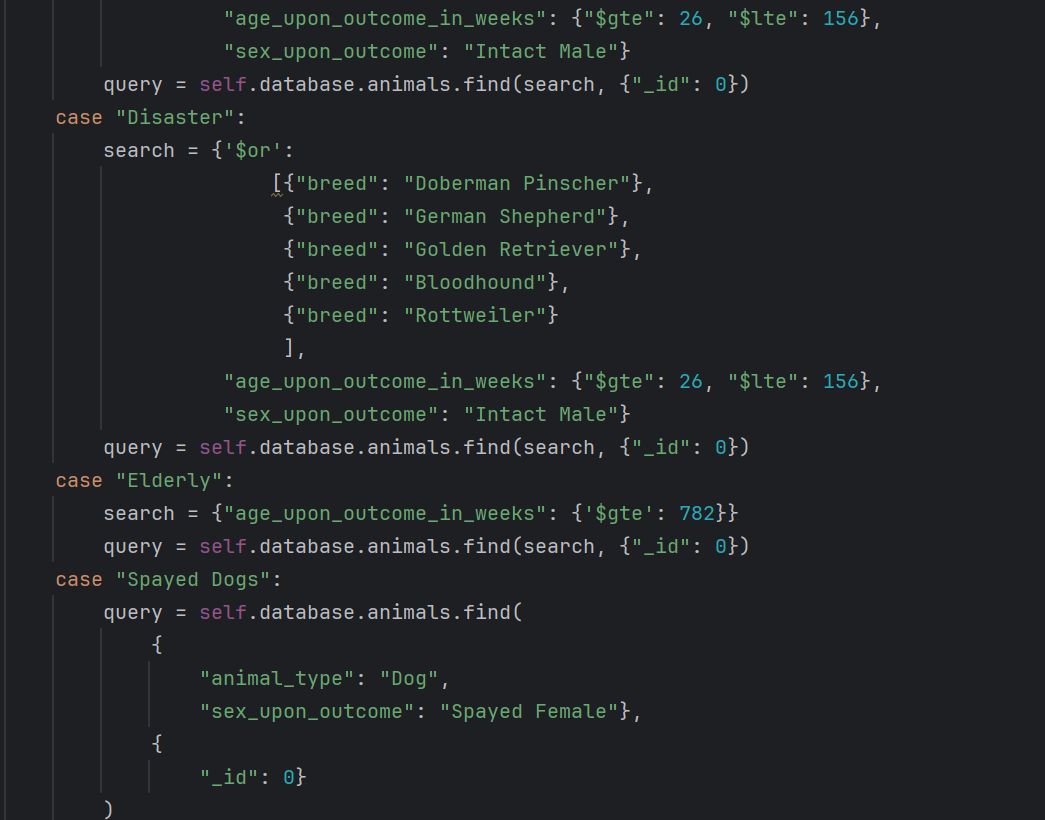


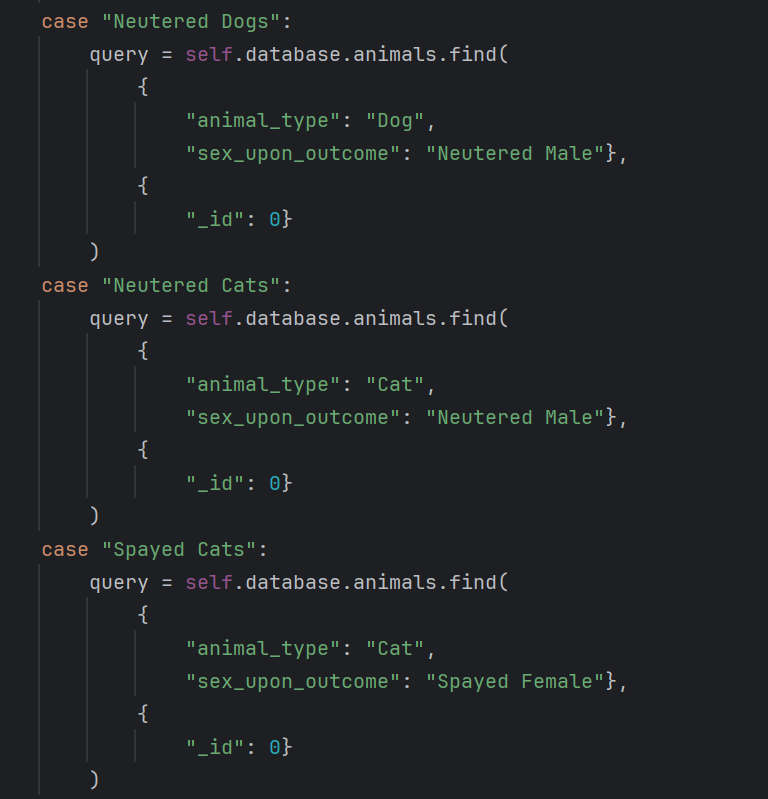


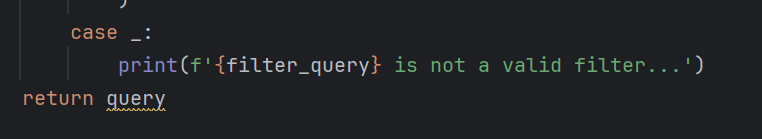
Creating the C.R.U.D. interface

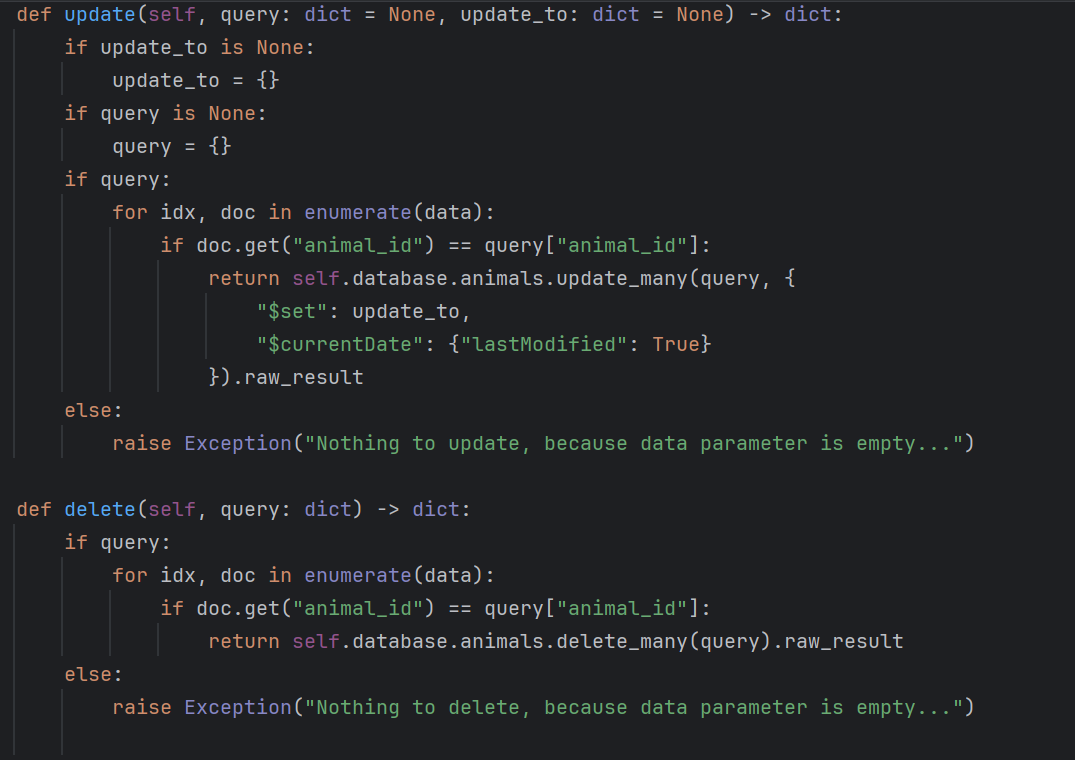






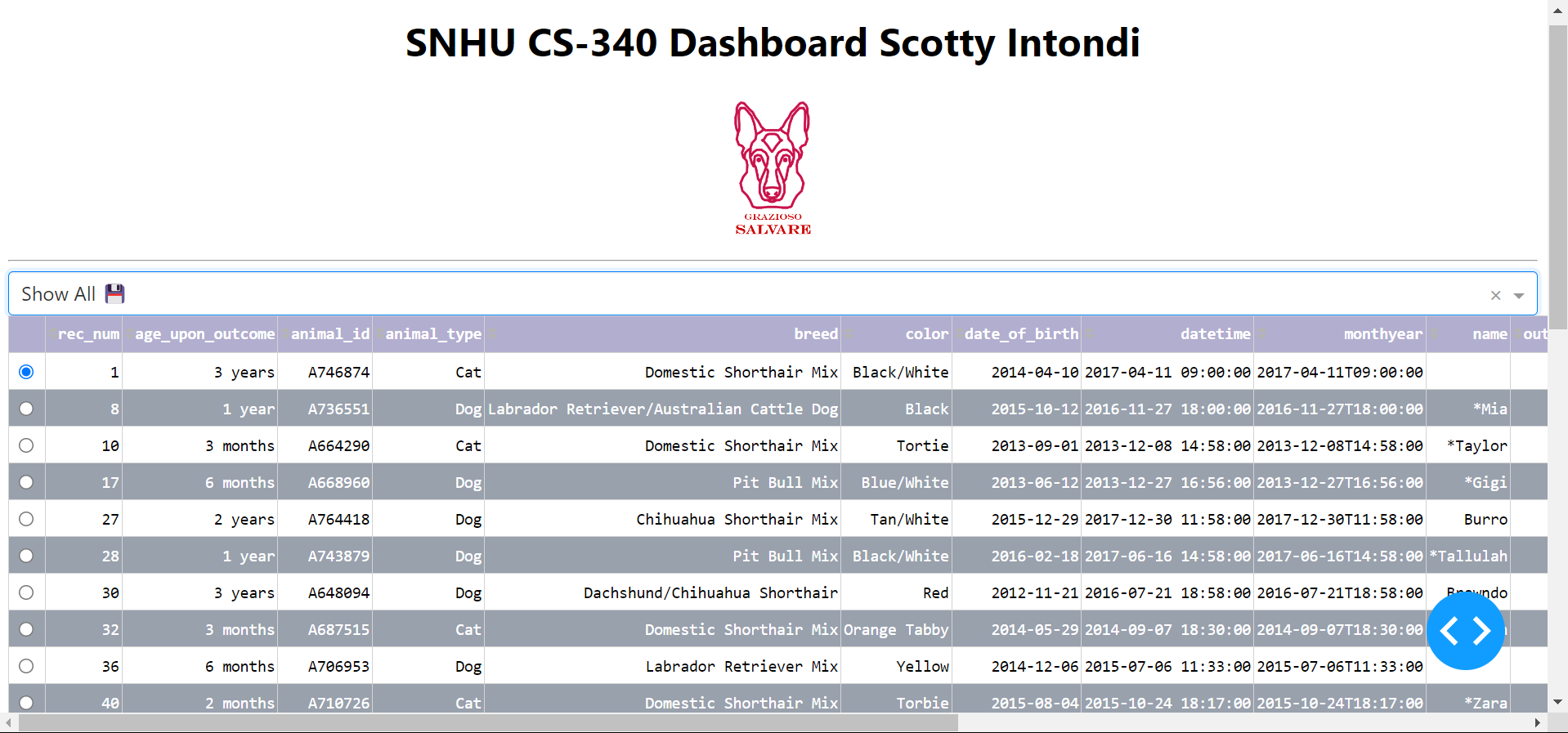


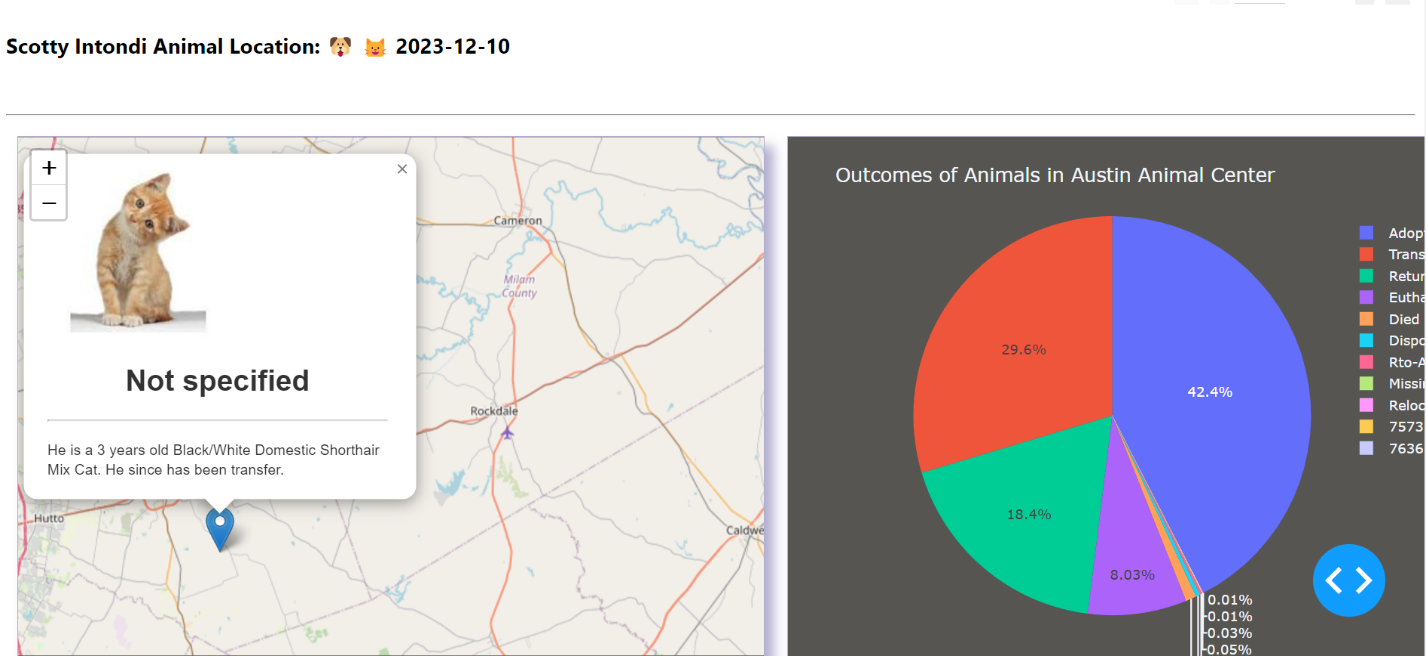




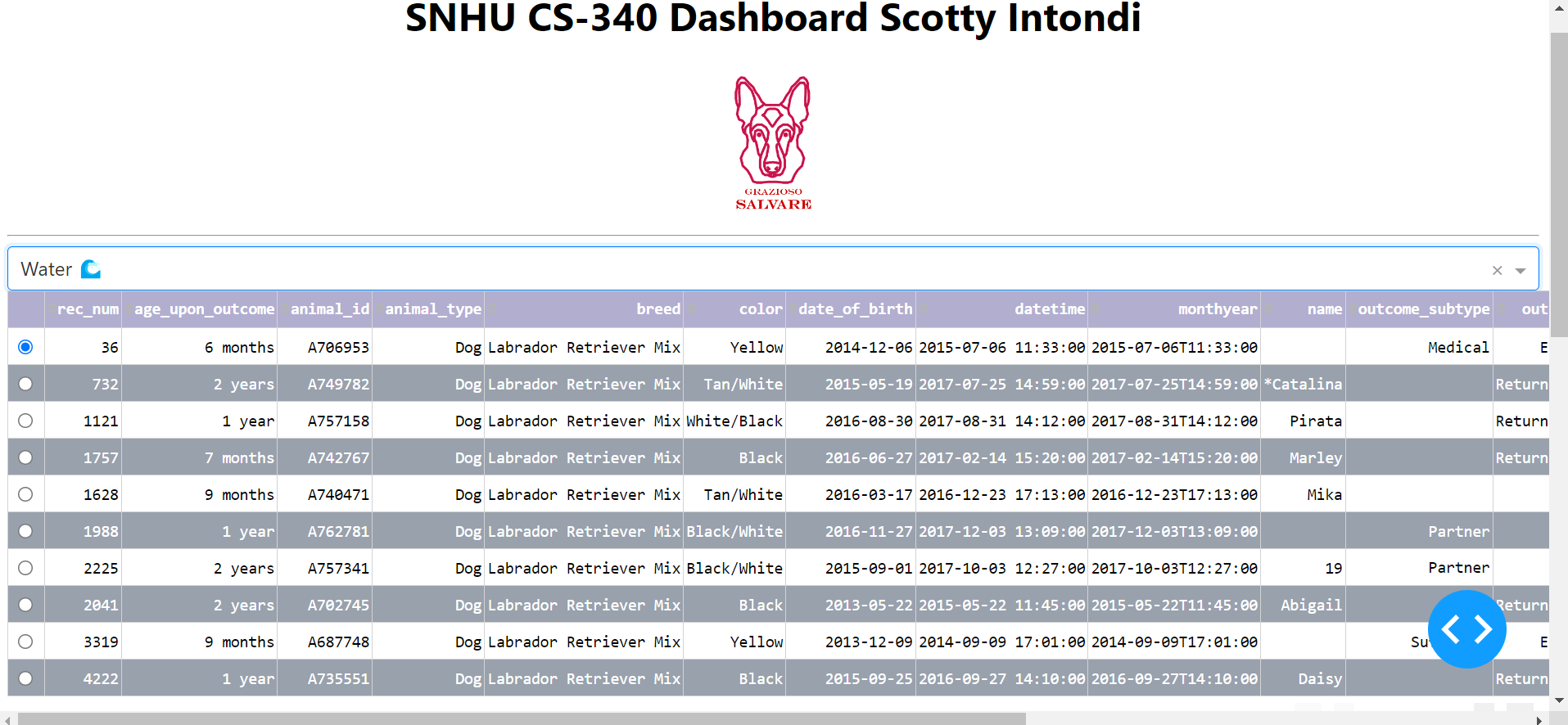
Dashboard UI for “Show All”, “Water”, “Mountain”, “Disaster”, and “Elderly”:

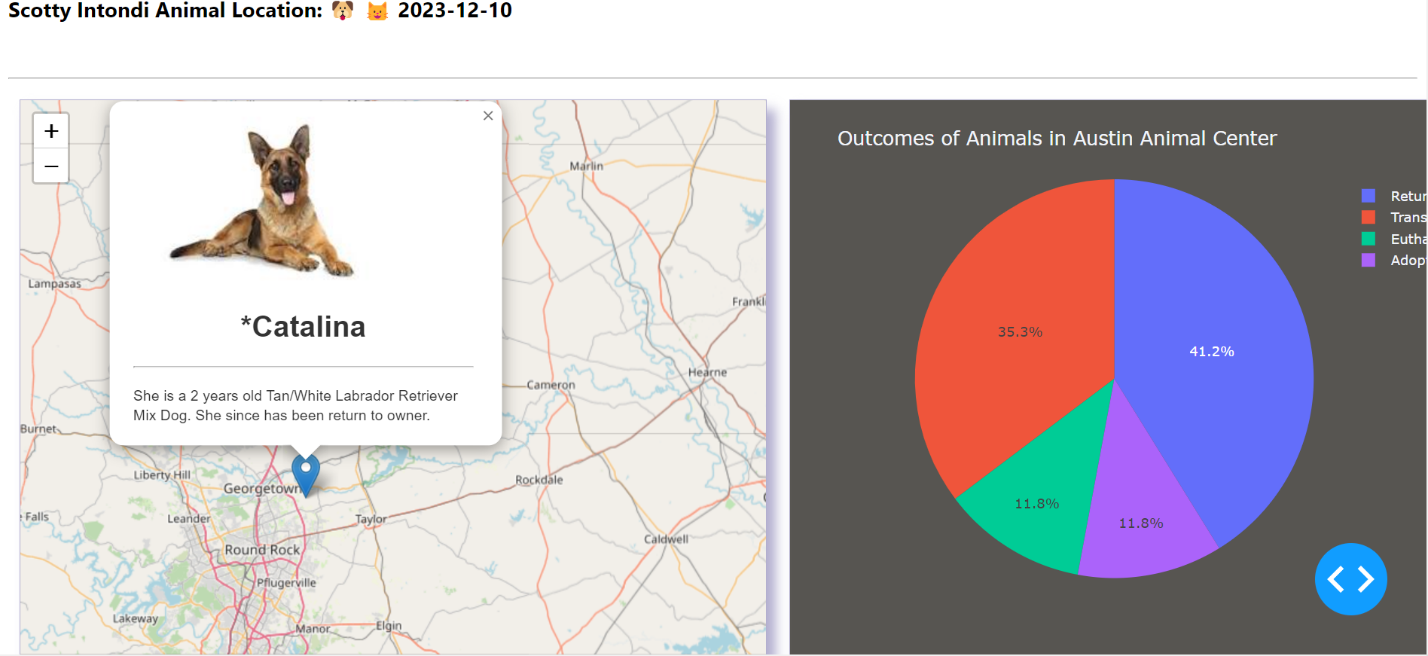
“Show All”:



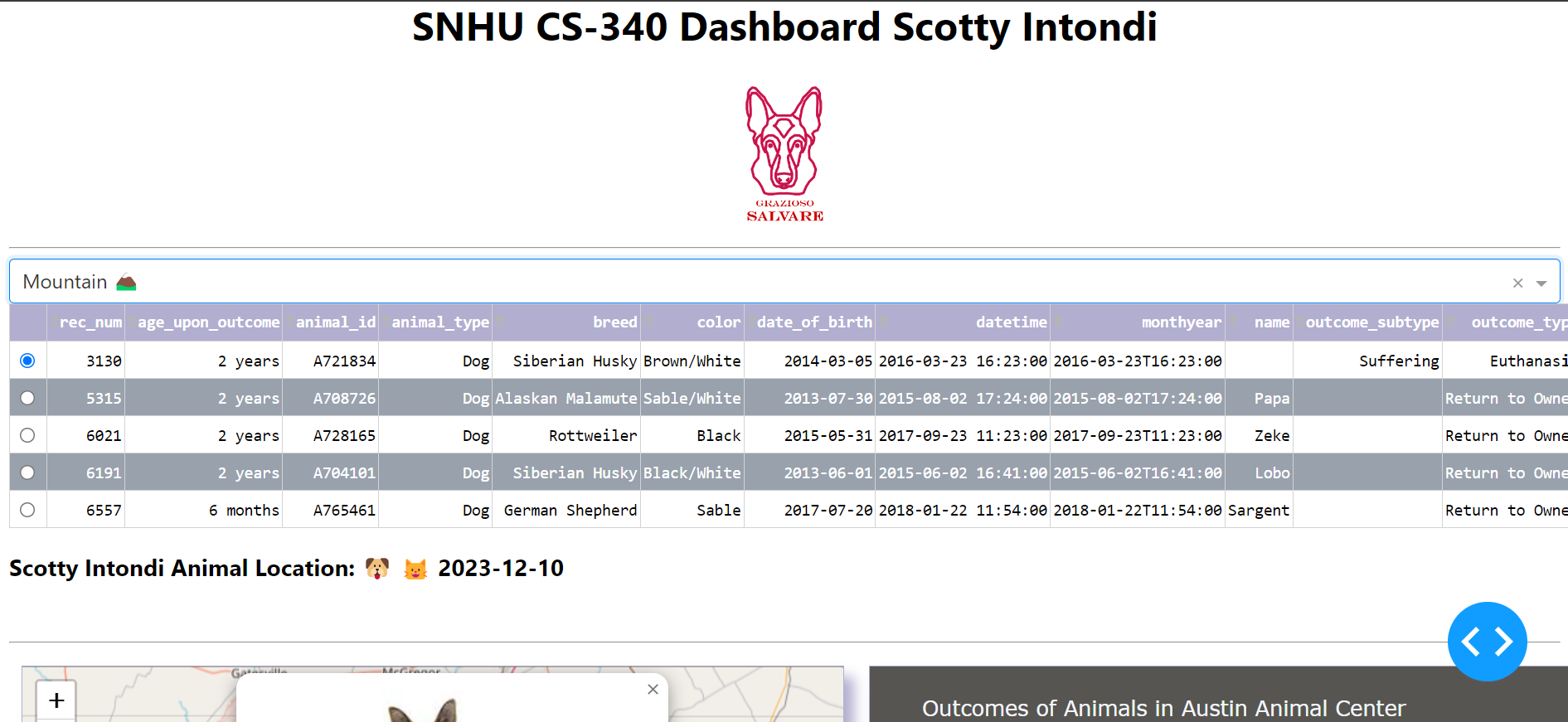


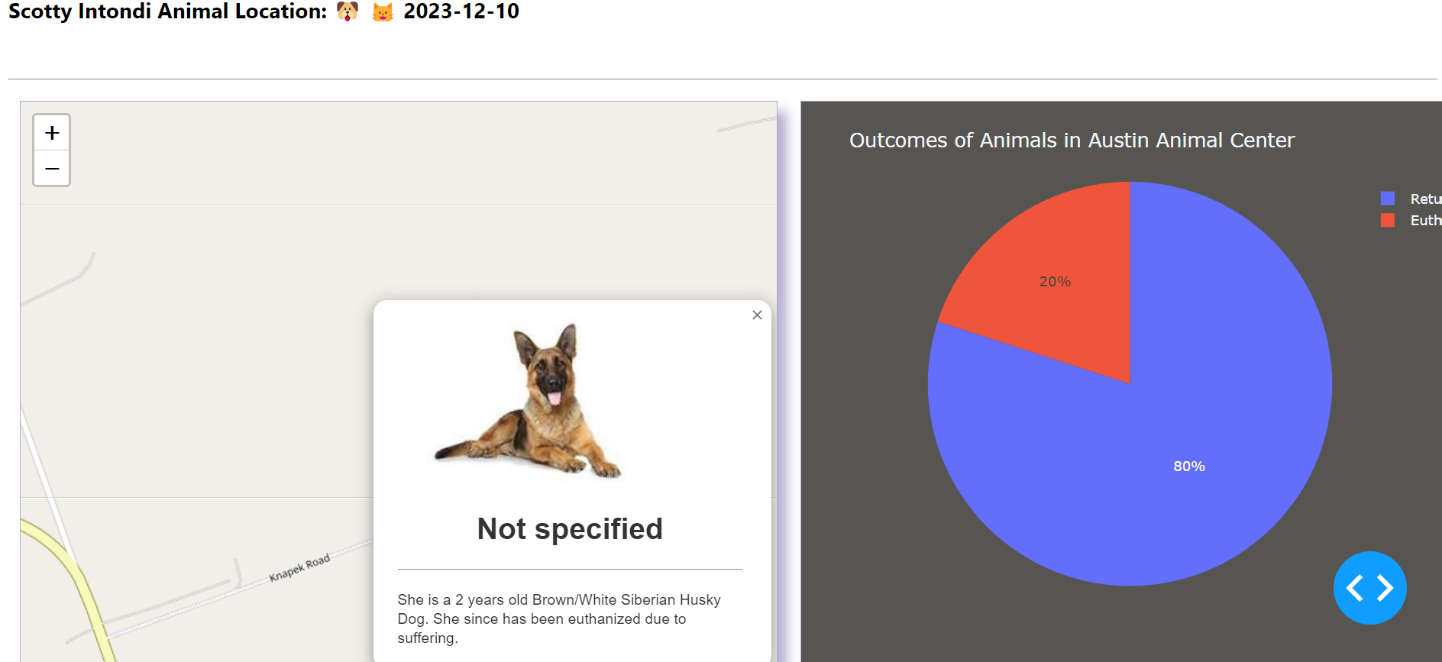
“Water”:



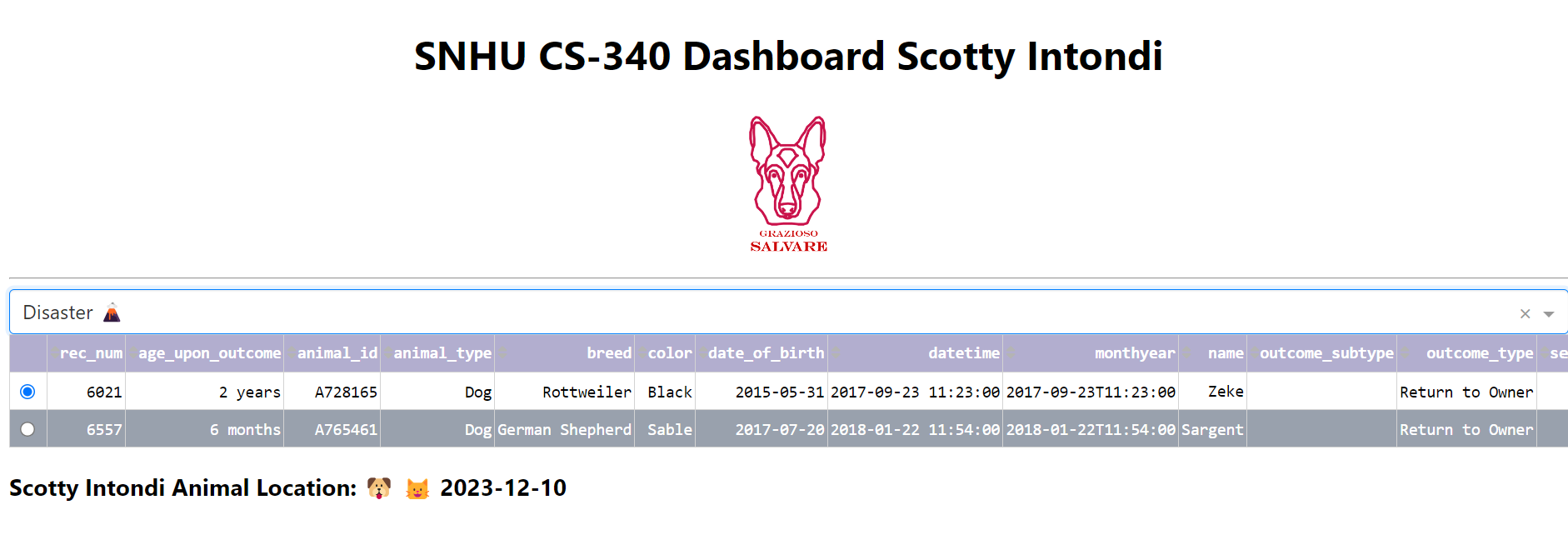


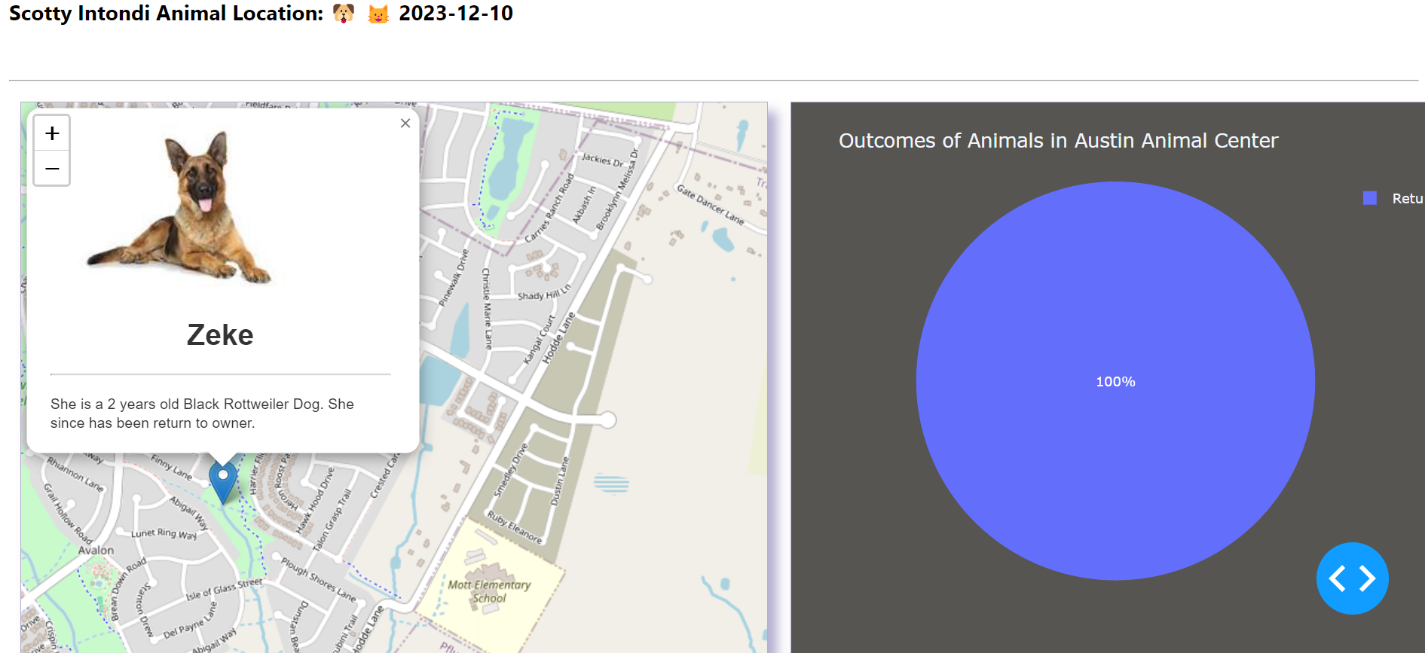
“Mountain”:



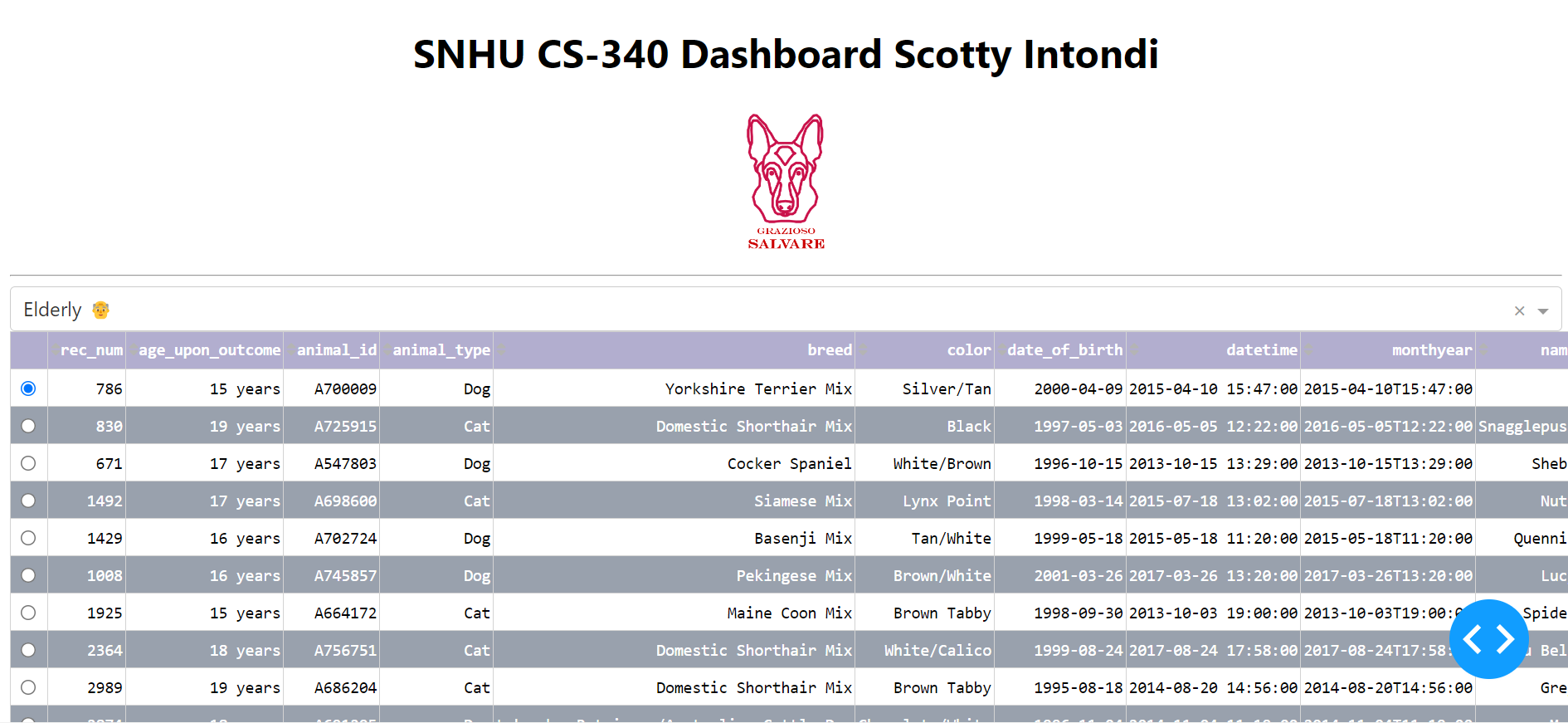


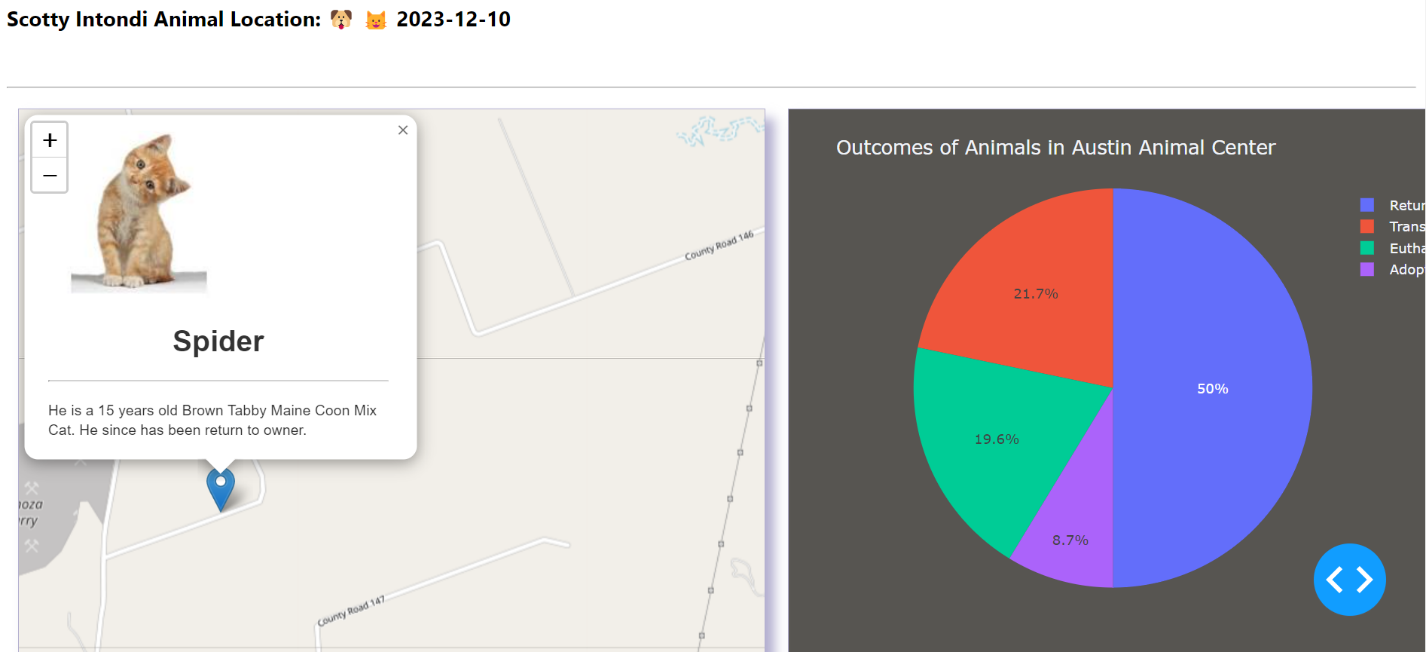
“Disaster”:





“Elderly”:





Challenges:

Working with the callbacks in Dash was a little different as although it’s made in JavaScript, you can’t grab the tags in Dash the same way through the DOM. You have to go through the children such as plotly etc… Also, callbacks are imminently called once the page is opened and I probably should’ve suppressed the initial nth click and be greater than 0 to keep from rendering on refreash.

Contact

Scotty Intondi

[intondiscott@snhu.edu](mailto:intondiscott@snhu.edu)

CS-340 Client/Server Development