# CS 340 README - MongoDB and Dashboard Application

## About pyCRUD and Dashboard

pyCRUD is a Python module allowing users of a MongoDB database to perform the **C**reate, **R**ead, **U**pdate, and **D**elete functions on documents in a MongoDB collection. Only Create and Read have implemented so far, but Update and Delete functionality is planned for the future. The dashboard project two is a user-friendly interface requested by Grazioso Salvare to easily search through the available data with a GUI.

## Motivation

This module began development because of the need for the rescue animal training company Grazioso Salvare to navigate through a database of animals to find ideal candidates for their training program. The Python module would allow scripts or other front-facing applications to interact with the database, providing a middle-layer and increasing accessibility to the data. Because of this, any type of application can make use of this module to access a MongoDB database, not just one for shelter animals.

## Requirements and Installation for pyCRUD

1. **MongoDB**: You will need an active MongoDB instance running on your server. Installation and startup instructions, depending on your OS, can be found here: <https://www.mongodb.com/docs/manual/installation/>
2. **PyMongo Driver:** Follow these instructions to install the driver that allows MongoDB to be controlled with Python functions (this will also provide dependencies such as pip and Python3 itself): <https://www.mongodb.com/docs/languages/python/pymongo-driver/current/get-started/>
3. **pyCRUD module:** The pycrud.py file will need to be downloaded and placed in whatever directory you choose.
4. **Your Data:** Finally, make sure to include your own datasets, ready to import into MongoDB.

## Getting Started

Now that you have everything installed, here’s how to get set up using pyCRUD with your database.

1. Once MongoDB is running, enter the shell with this command:

> mongosh

1. Import your dataset as a MongoDB database using the mongoimport command, which will vary in syntax depending on your MongoDB configuration. Details on mongoimport can be found here: <https://www.mongodb.com/docs/database-tools/mongoimport/>
2. Make sure to create a user with Create and Read permissions that the pyCRUD module can use to authenticate. Update and Delete will also be required with future versions.
3. You are now ready to create scripts or applications that import the pyCRUD file and use its functions to interact with your MongoDB database.

## Usage

The pyCRUD.AnimalShelter class has several member variables that are required for connecting to the MongoDB Client. These will have to be changed to match your MongoDB instance. The usable functions are AnimalShelter.create(data), and AnimalShelter.read(query).

**AnimalShelter.create(data):** This accepts data in the form of a Python dictionary and inserts it into the database as a document. The function outputs a Bool confirming success of the insertion.

**AnimalShelter.read(query):** This accepts a valid Mongo query and returns a list containing all documents that match the query.

### Code Example

# Import the pycrud module.

from pyrcud import AnimalShelter

# Create an instance of the AnimalShelter class, a simple Python dict, and a query.

myPycrud = AnimalShelter()

myDoc = {“key1”: “value1”, “key2”: “value2”, “key3”: “value3”}

myQuery = {“key1”: “value1”}

# Use the AnimalShelter class functions to add the document to the database collection, and then search for the given query.

successStatus = myPycrud.create(myDoc)

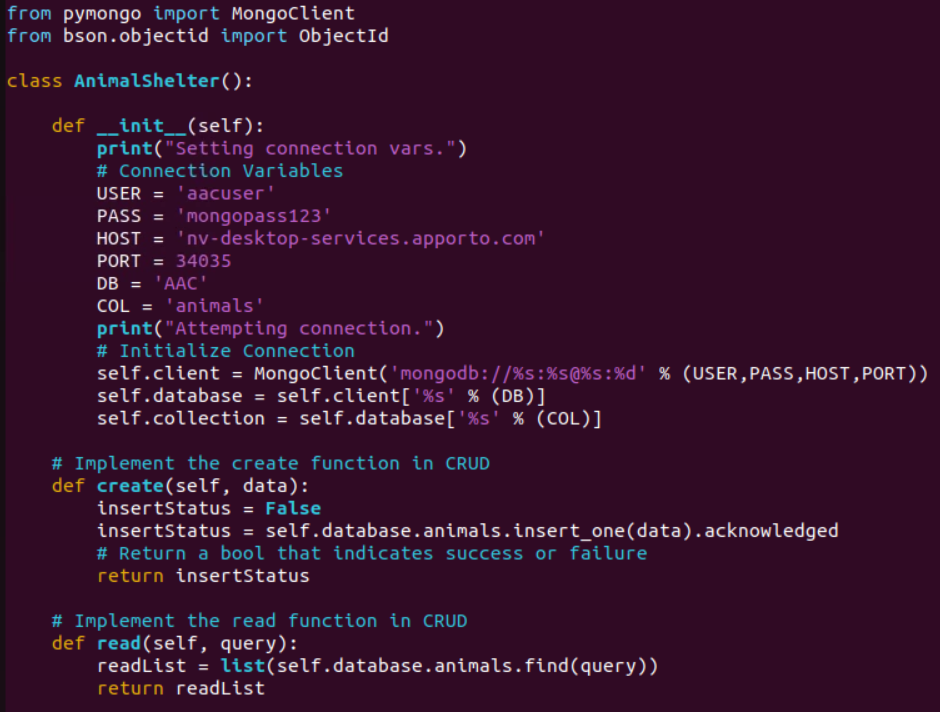
queryResults = myPycrud.read(myQuery)

### Tests

The pycrudTest file provides some basic testing for the module, and can be modified as needed to further test functionality. It consists of sections adding various documents, reading different queries, and then a primitive delete function clears the added test docs.

### Screenshots

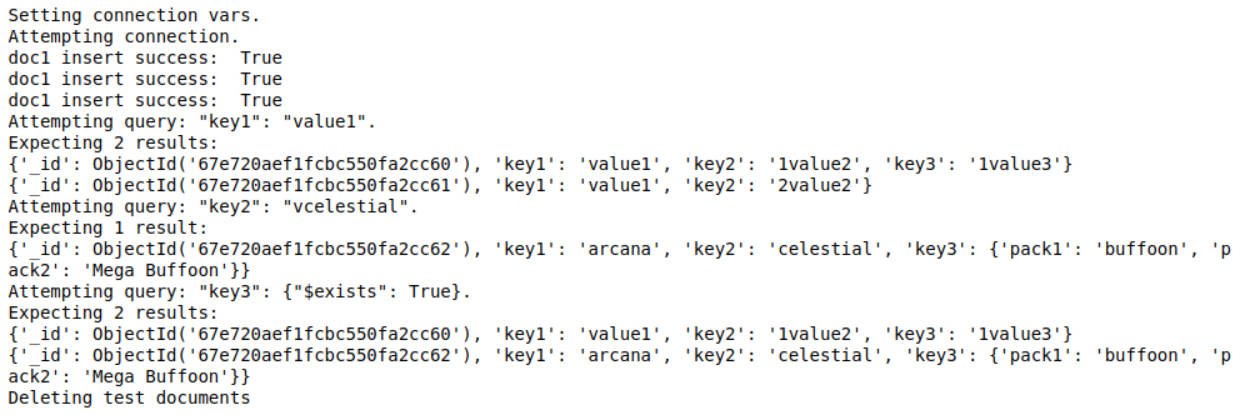
**(1/3) pycrud.py file:**



**(2/3) pycrudTest file:**

****

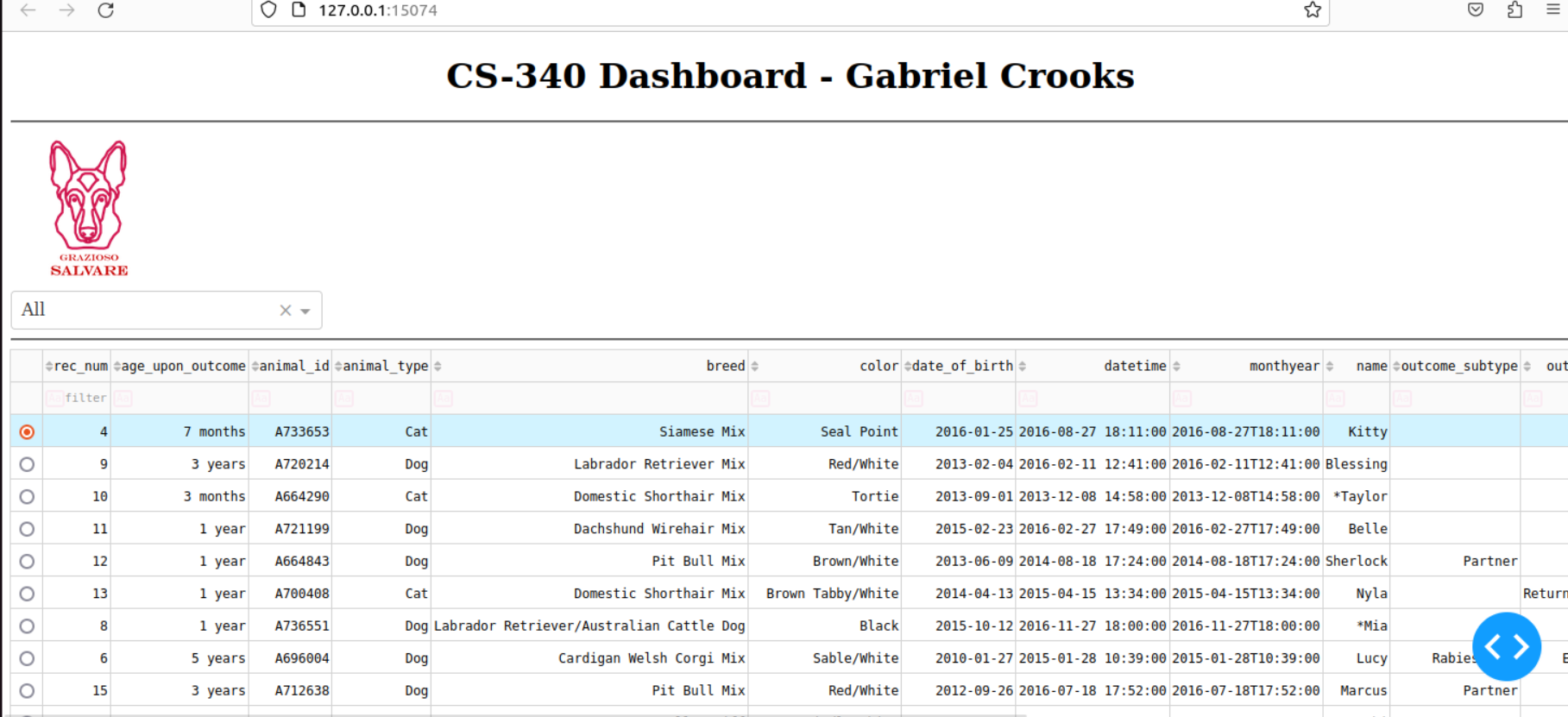
**(3/3) pycrudTest output:**

****

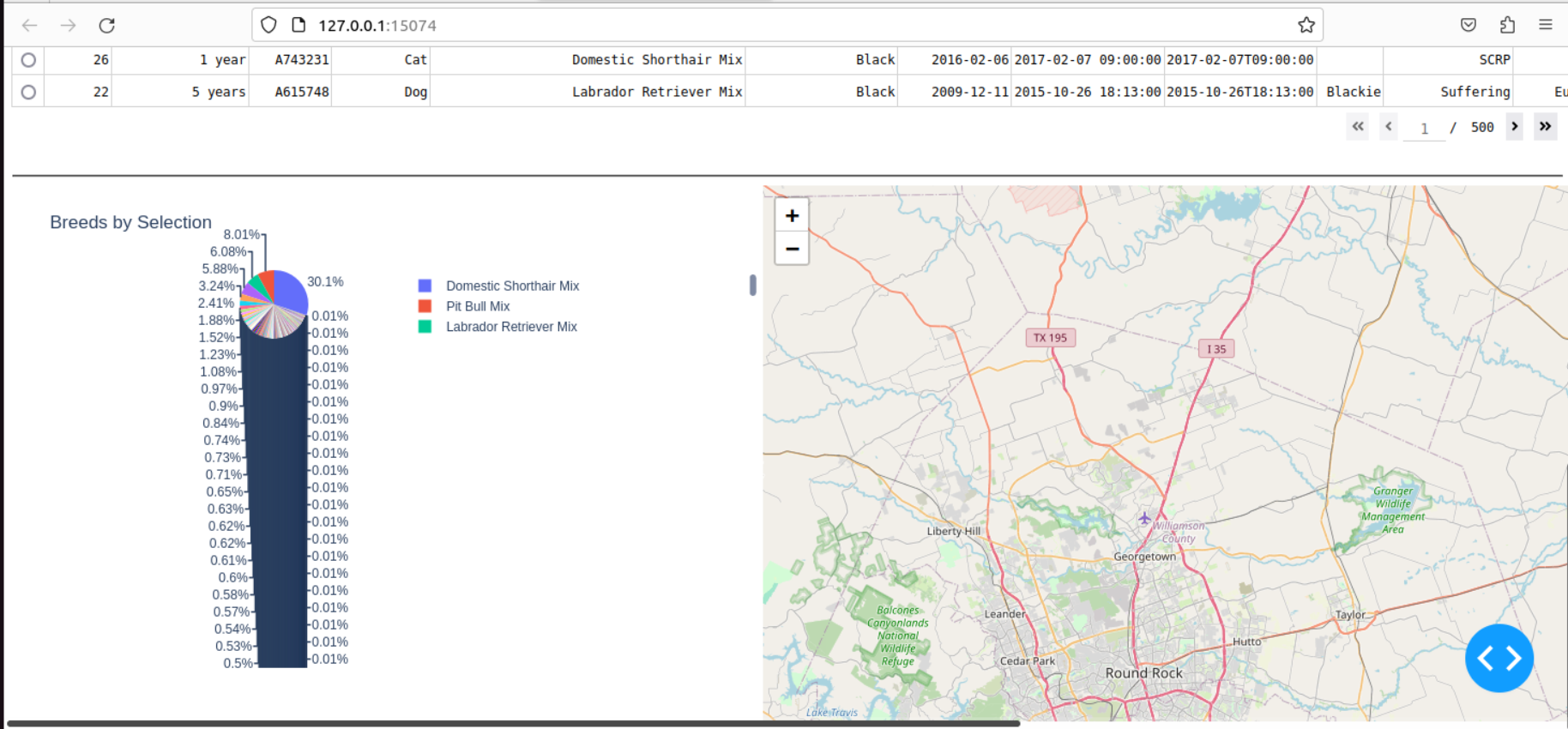
## Dashboard

The dashboard app displays the data from MongoDB in a user-friendly way. There are a variety of filtering options including sorting by column, search filtering on each column, as well as a few specific query filters requested by Grazioso Salvare: animals ideal for Water Rescue, Mountain Rescue, and Disaster Rescue. The app uses Python and Dash to display the currently filtered data in a table. Below it, a pie chart shows currently selected breeds and a map of the area. Callbacks create custom MongoDB queries using PyMongo upon filtering from the custom dropdown menu.

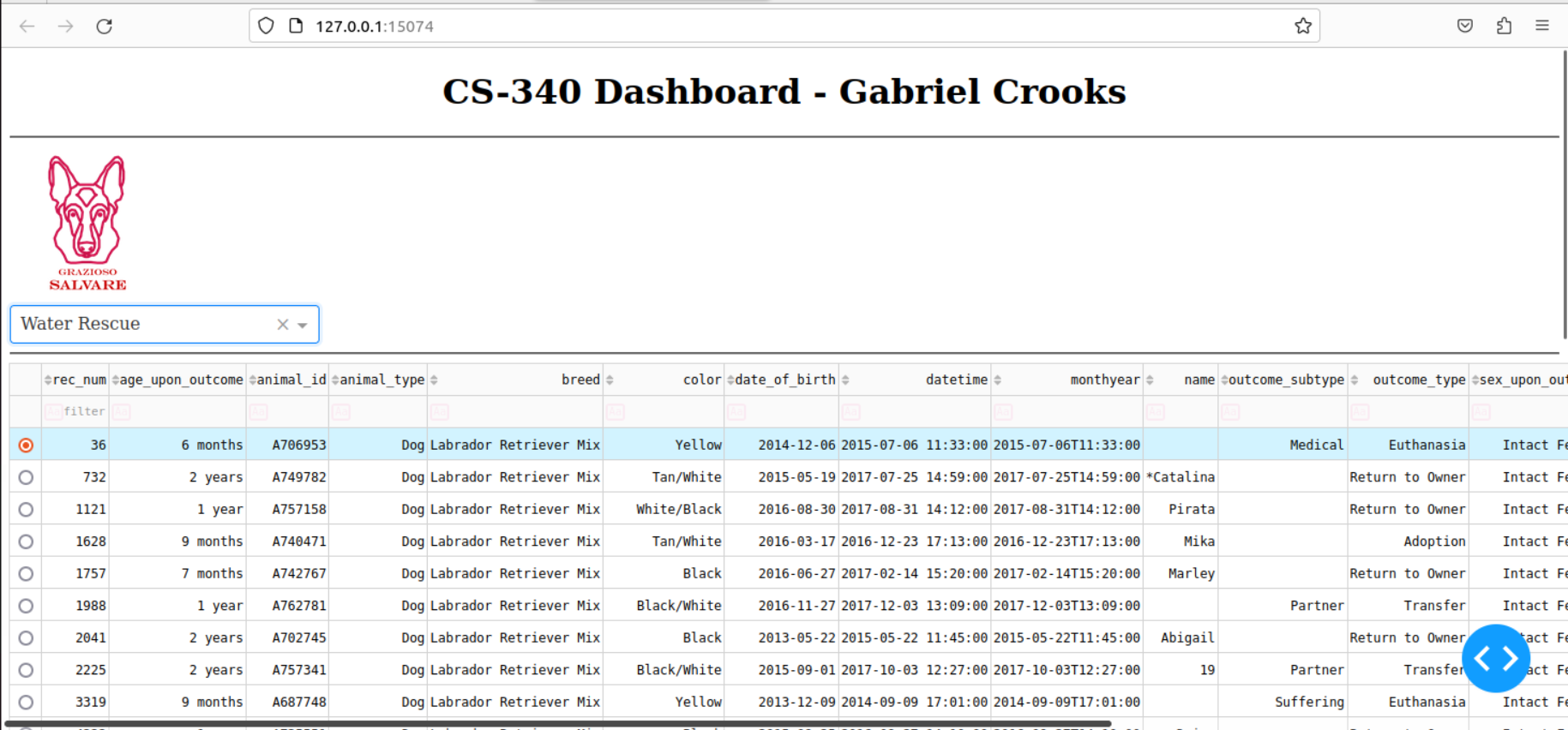
**Dashboard start (1/2):**

****

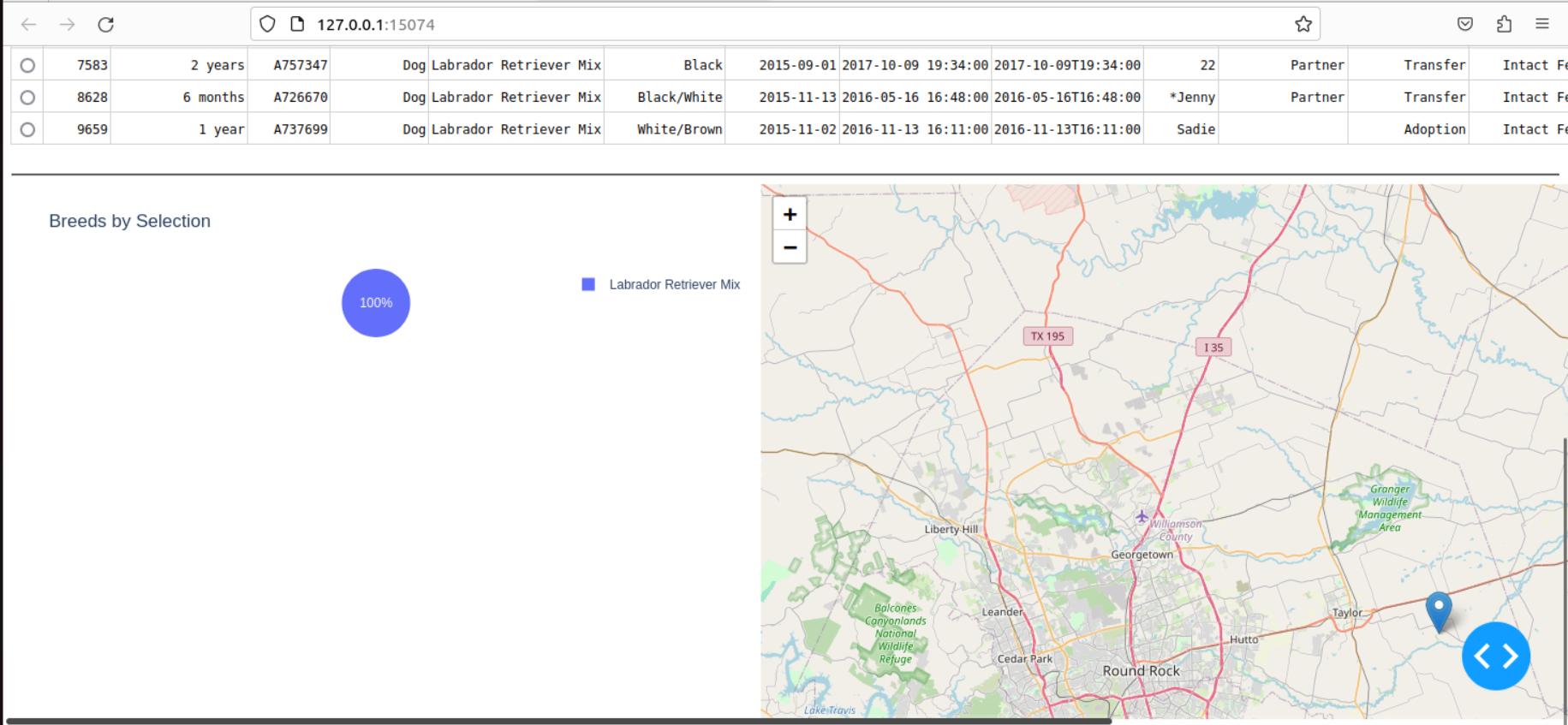
**Dashboard start (2/2):**

****

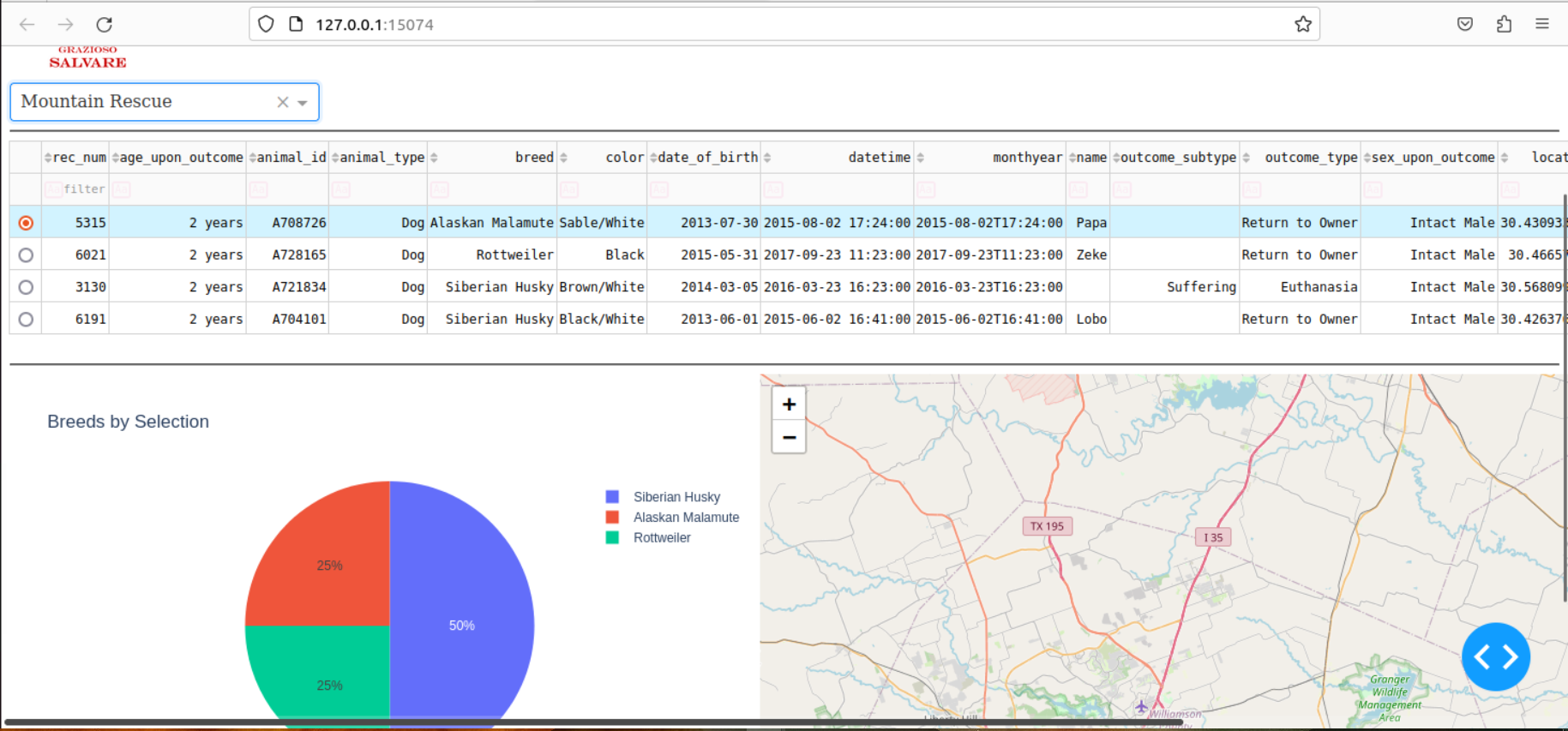
**Filter for Water Rescue (1/2):**

****

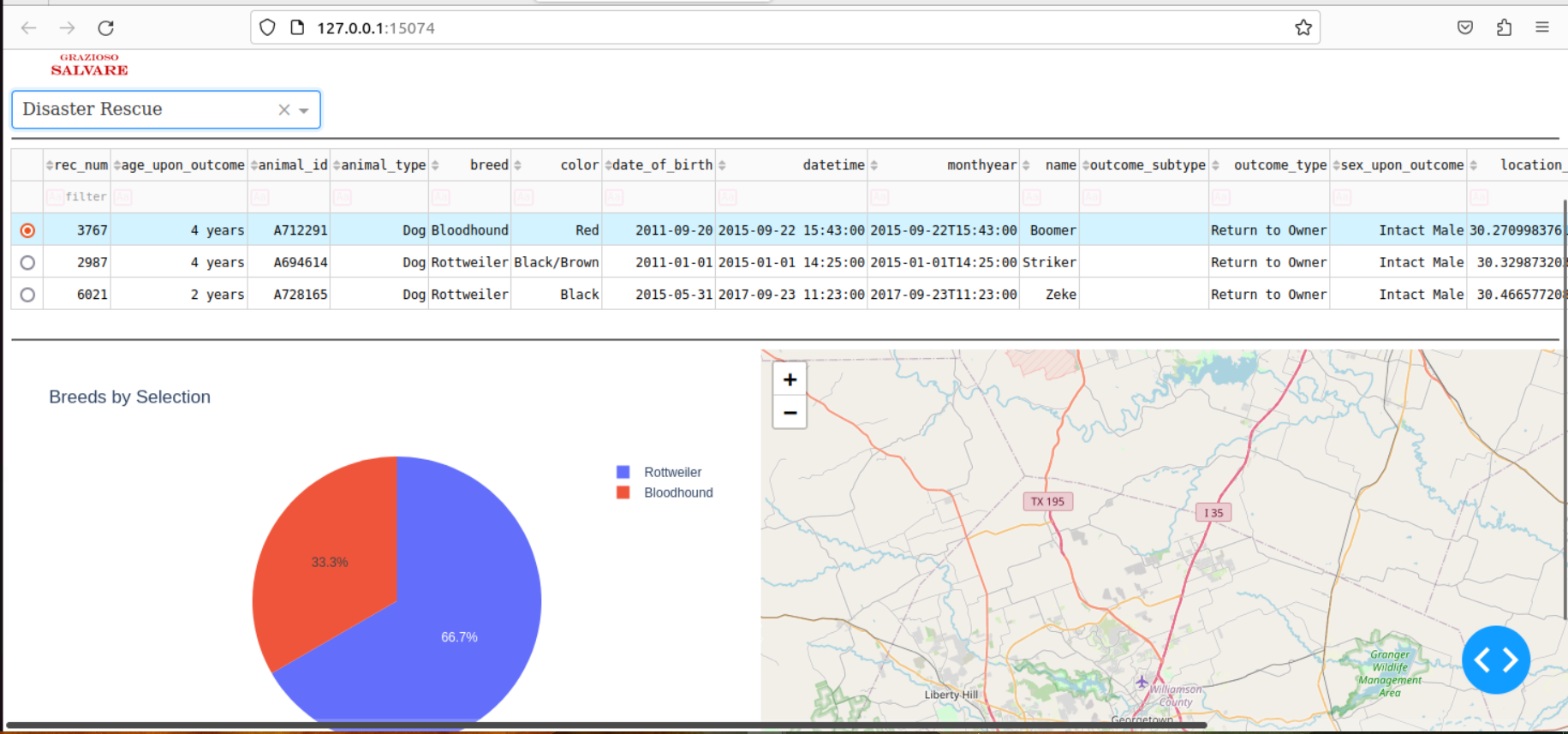
**Filter for Water Rescue (2/2):**

****

**Filter for Mountain Rescue:**

****

**Filter for Disaster Rescue:**

****

## Contact

Gabriel Crooks, [gabriel.crooks@snhu.edu](mailto:gabriel.crooks@snhu.edu)