# CS 340 README

## About the Project/Project Title

The project utilizes a MongoDB database that lists dogs available at an animal shelter. A module written in Python is used to collect data from the database, which is then displayed in an intuitive dashboard interface that provides the animals in a listed format, along with a pie chart to detail the number of breeds and a geolocation tool to map their locations.

## Motivation

The project exists to allow Grazioso Salvare to filter through a database of dogs that are available for training. As training for search-and-rescue dogs is more effective in dogs under two years of age and certain types of dogs are better in certain environments, this project aims to set up a filter that will list available dogs that meet specific criteria: age along with their respective environment.

## Getting Started

To get started, the csv file with database information needs to be downloaded and imported into MongoDB. The data should be indexed for faster load times, and a user account with read and write access should be used. Python will need to be installed for the program to run. The dashboard is written as a .ipynb file and was created with Jupyter Notebook.

## Installation

MongoDB is required.

Python is required and can be downloaded from Python’s website (<https://www.python.org/>)

Installation instructions for Jupyter Notebook can be found on their website (<https://jupyter.org/install>

PyMongo, numpy, pandas, matplotlib.pyplot, and ObjectId libraries in Python are used.

The Jupyter version of the Dash framework is used for the dashboard interface.

## Usage

The following sections detail the usage of the AnimalShelter class as well as the Dashboard .ipynb file.

### Code Example

MongoClient from the PyMongo library is used to initialize a connection to the database. For security purposes, the personal information from this code block has been omitted. Data is taken as dictionary input and passed into a create method. This method only performs the create operations if input is in dictionary format and not empty. Data can also be passed in to a read method that will search through the database and return any items that exist in the database which meets the criteria. The update method takes an old document with a new parameter to update the document with, and then will check that both parameters contain a non-empty dictionary. It will then update the old entry with the new and return the number of updates. The delete parameter takes a dictionary parameter, validates that the parameter has data and is a dictionary, and then deletes every entry that matches the parameter. The number of deleted documents is returned.

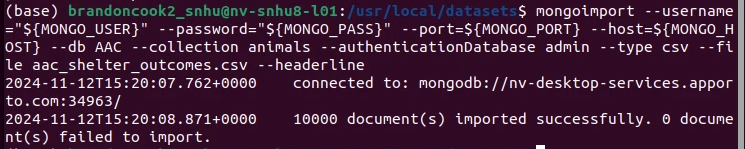
The dashboard file uses HTML code to create a dashboard to view data, which is filtered by radio buttons and updates in a table of rows and columns. There is a pie chart and geolocation widget to display locations of animals. The screenshots have been zoomed out to capture all relevant information.

### Tests

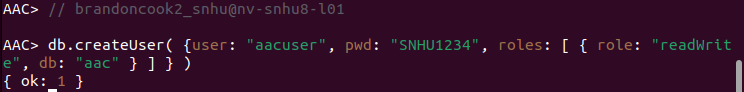
The test procedure first creates a new animal shelter object, then clears the database before testing. The create method is tested with a sample entry, which is read using the read method. The entry is then updated to test the update method, and read from to verify the results and changes. Then, the delete method is testing to delete the entry that was just updated, and then the results are displayed. Because there are no results, there are no further entries following “Testing Delete Method.”

### Screenshots

Importing CSV file:



Creating user account:

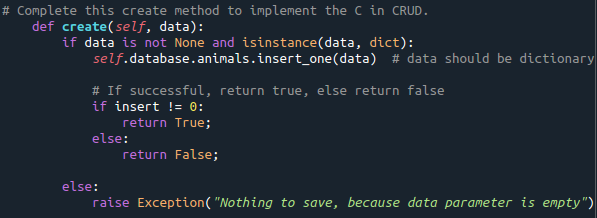


Initializing connection:

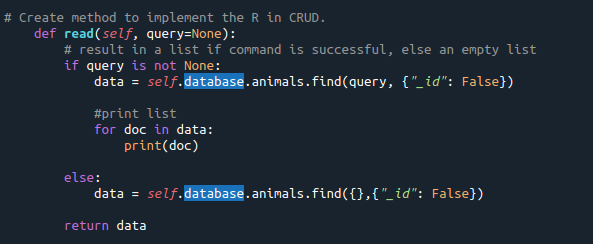
A computer code with green and white text

Description automatically generated

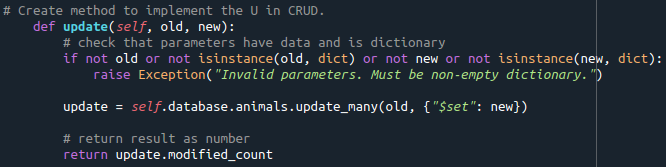
Create method:



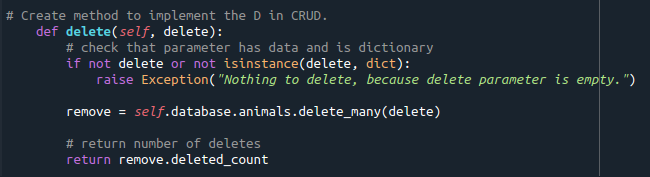
Read method:



Update method:



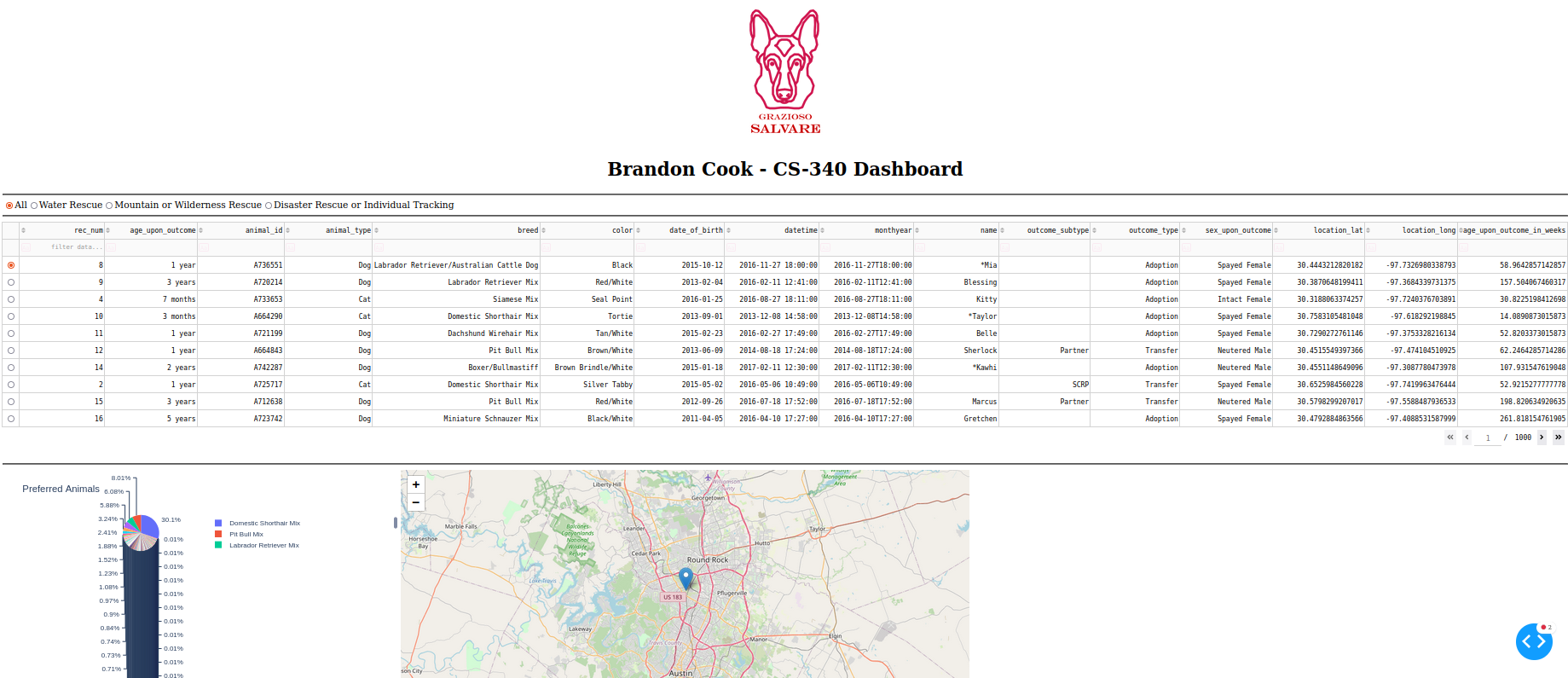
Delete method:

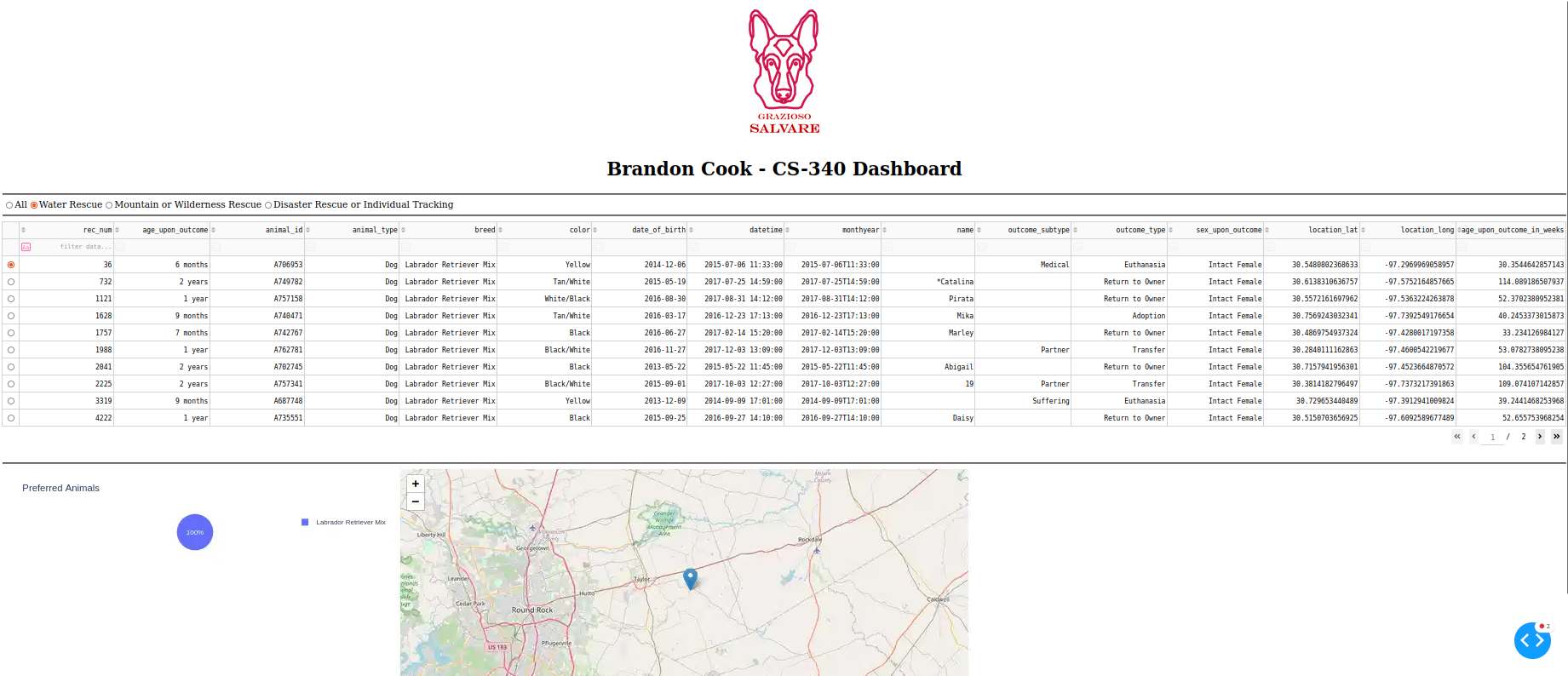


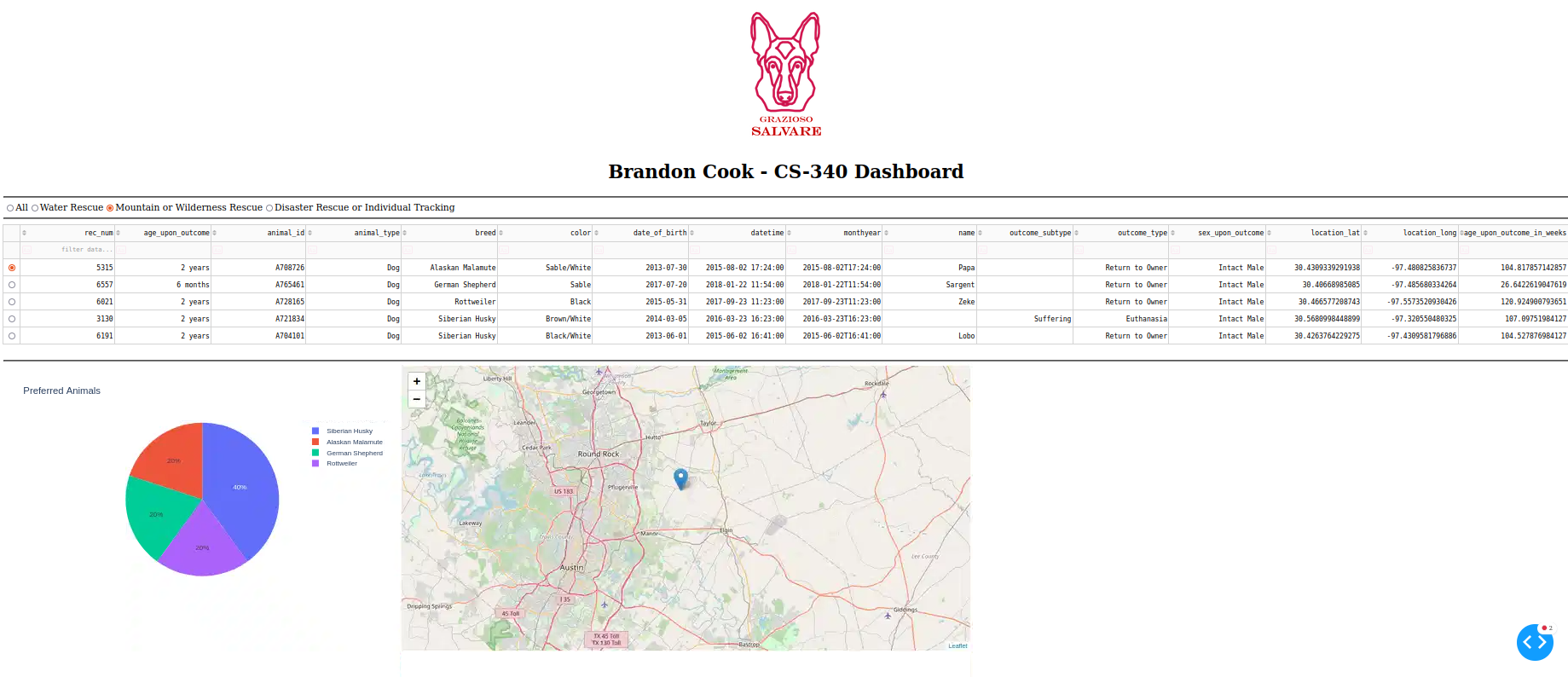
Test:

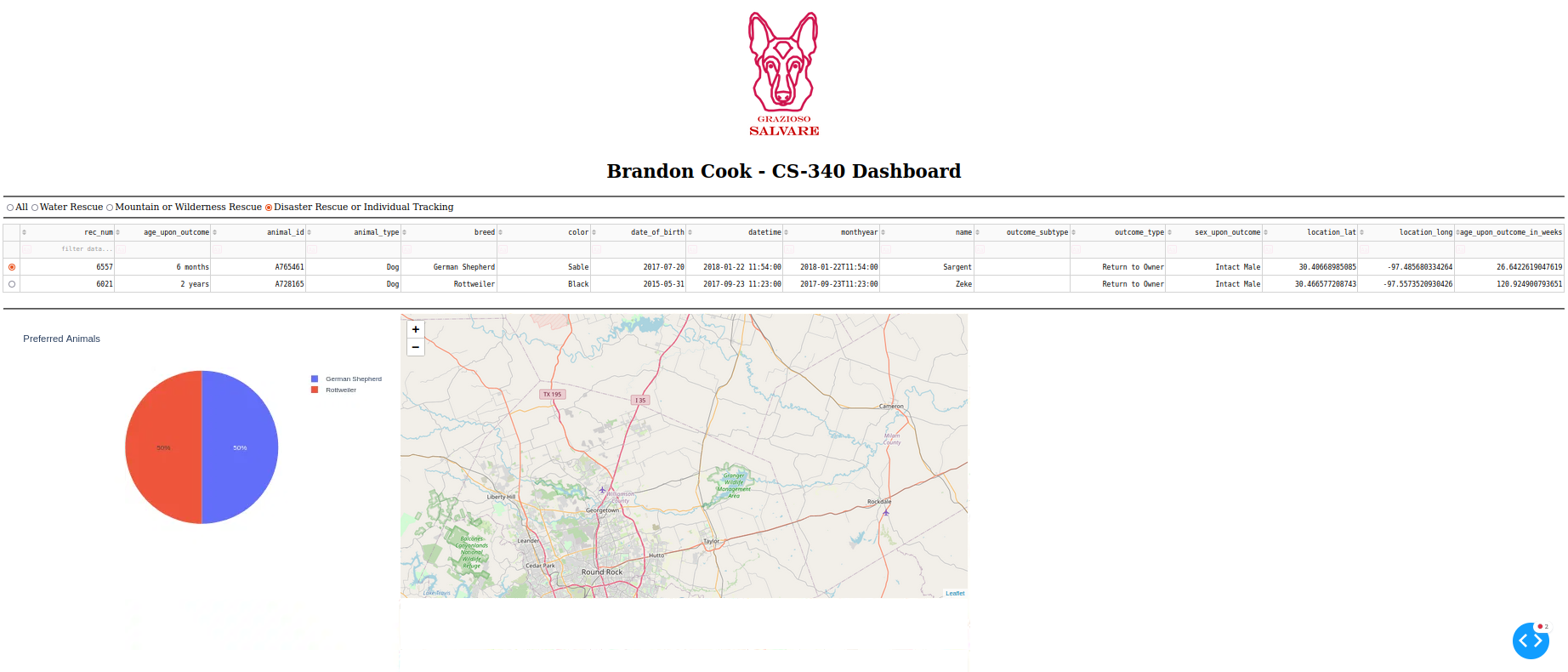


Dashboard:









## Contact

Name: Brandon Cook