# CS 340 README Template

## About the Project/Project Title

CRUD.py is intended to allow for create, read, update, and delete operations to be used with mongodb.

## Motivation

This was made to use CRUD operations more easily in mongdb with different applications. The dash frontend enhances the usability of the app and can allow for predetermined filters to be set. Also, because I am getting a grade on this and would like to graduate.

## Getting Started

First thing is to setup a mongodb database, which information on how can be found on their website: [www.mongodb.com/docs/manual/](https://www.mongodb.com/docs/manual/). After the data base is set up you can use the library by writing the line: from CRUD import animalShelter (the class name animalShelter can be changed locally to be something more general). Then instantiate the class by calling animalShelter(user, pass), where user and pass are the username and password to login to the database you created earlier.

The dash framework allows for a frontend to be build allowing users to more easily use the system. Dash allows for html components to be used using python code allowing us to combine pymongo with a frontend making a more rounded app. Dash installation instructions here: <https://dash.plotly.com/installation>

## Installation

The only libraries needed are pymongo and bson.objectid both of which can be installed using the single python command “pip install pymongo”. Use the mongoimport tool to import the database, example below.

**NOTE: Pymongo states that using pip install bson will install an incompatible version of bson and the user should just use the bson library that is included when installing pymongo.**

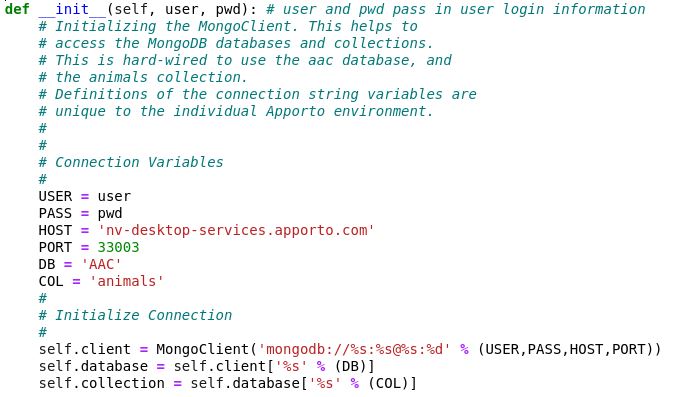
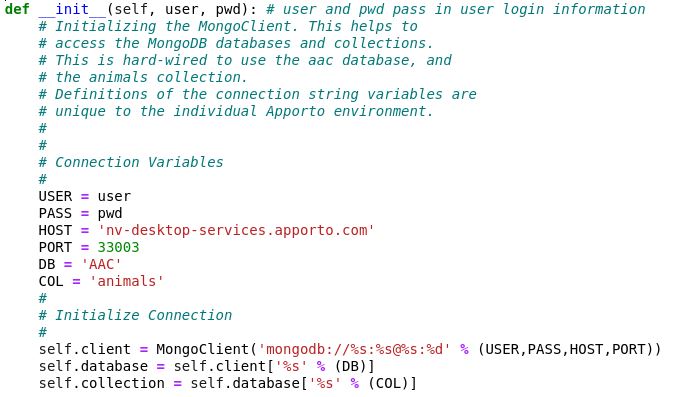
A screenshot of a computer screen

Description automatically generated

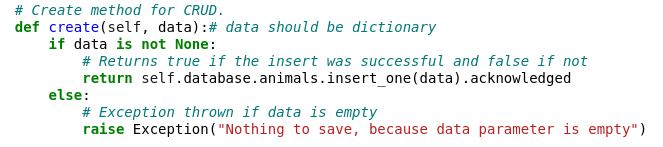
## Usage

### Code Example

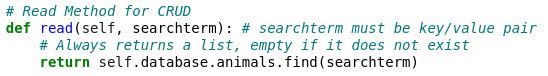
Constructor: takes two arguments (username, password) that is used to login to the database.



Create method: takes a set of key/value pairs as the argument and returns true if the data was successfully inserted into the database and false if not.



Read method: takes a key/value pair and returns a list of key/value pairs of what it finds, or an empty list if it doesn’t find anything matching



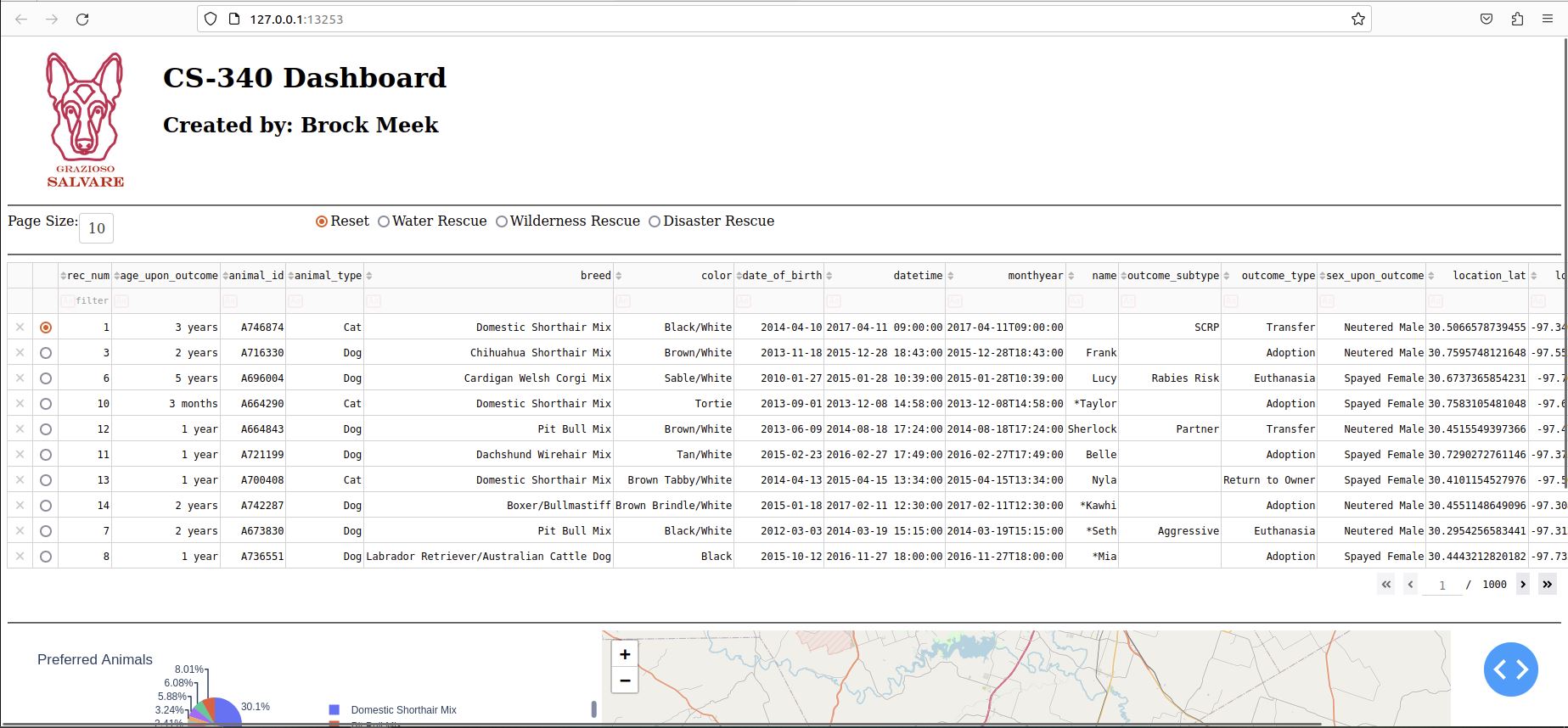
Update method: takes a query to find the element and the updated key/value pair as arguments and returns the amount of elements that were modified.

A white screen with black text

Description automatically generated

## Dashboard Functionality:

Opening Page: The opening page shows the Grazioso Salvare logo along with the CS-340 Dashboard title and my accreditation below that. It automatically shows all entries in the database along with a pie graph showing percentages of breeds and a map of where the animals are housed.



Page Size: The drop down right below the logo on the left side of the page gives the user the ability to change how many entries are shown on the page at a time. The current options are 5, 10, 15, 20, 30, 40, 50, and 100.

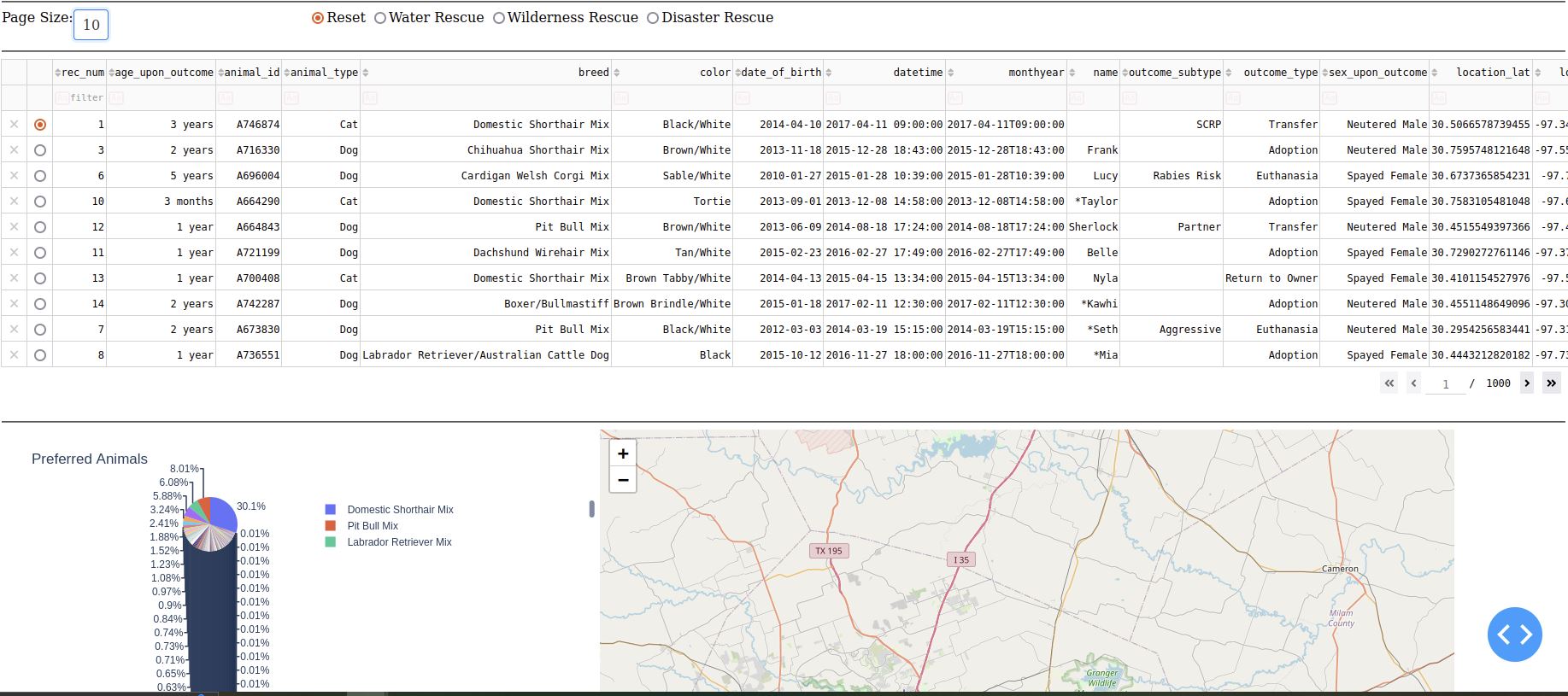
A screenshot of a computer

Description automatically generated

Filter Options: Next to the Page Size dropdown are a set of filter options the user can cycle through one at a time. The filters will filter the list based on breed, sex, and age predetermined by Grazioso. The filters also change the data shown on the pie chart and map.



Reset filter:



Water Rescue filter:

A screenshot of a computer

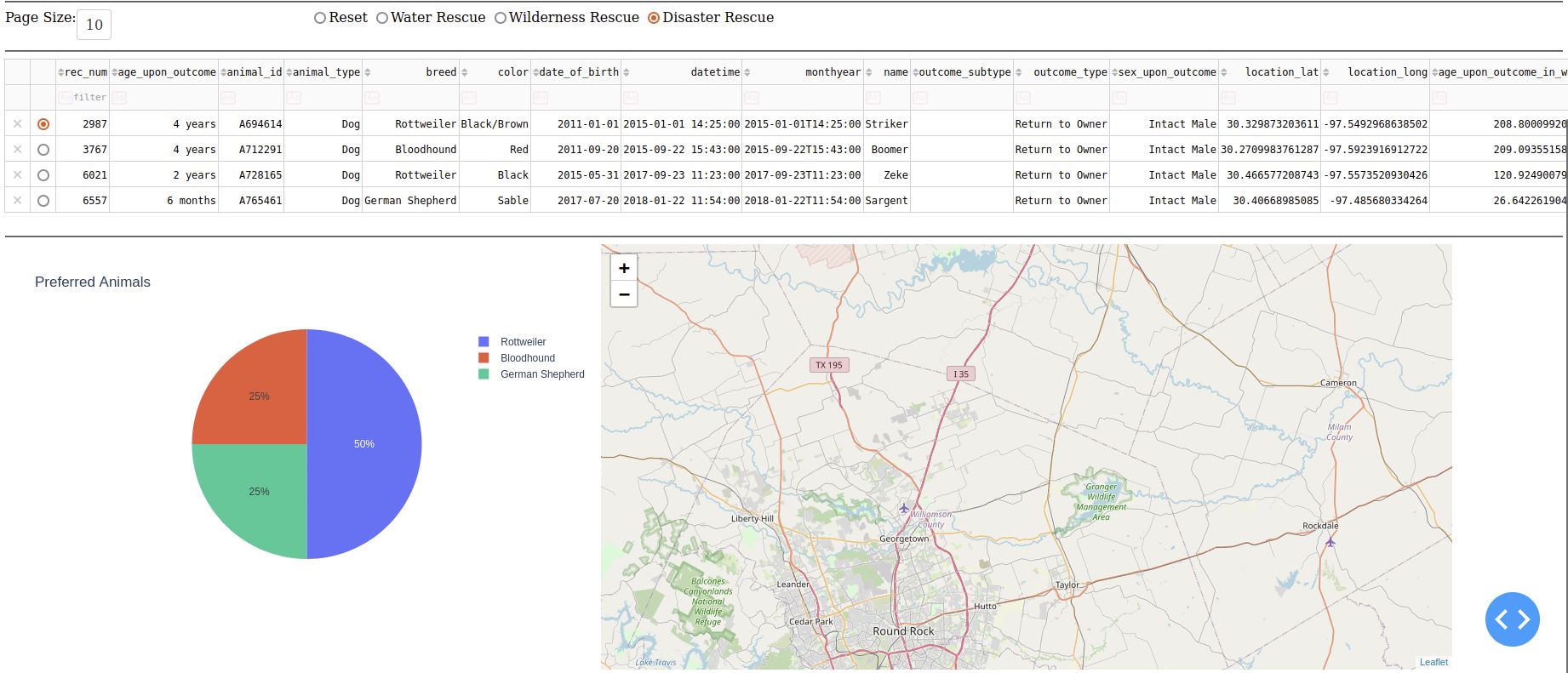
Description automatically generated

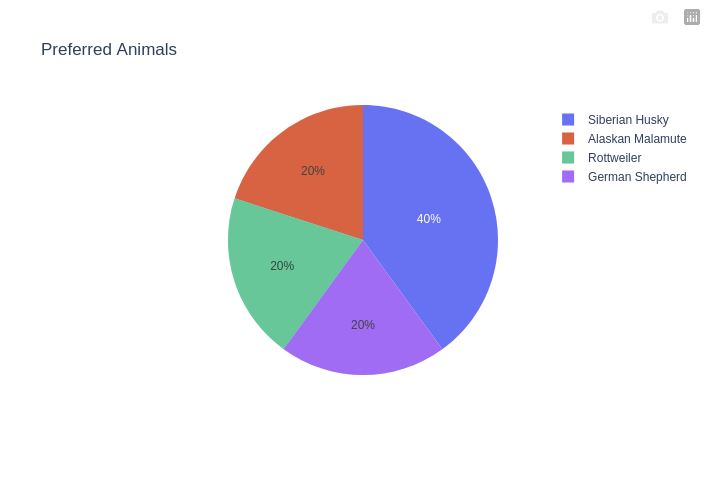
Wilderness Rescue filter:

A screenshot of a map

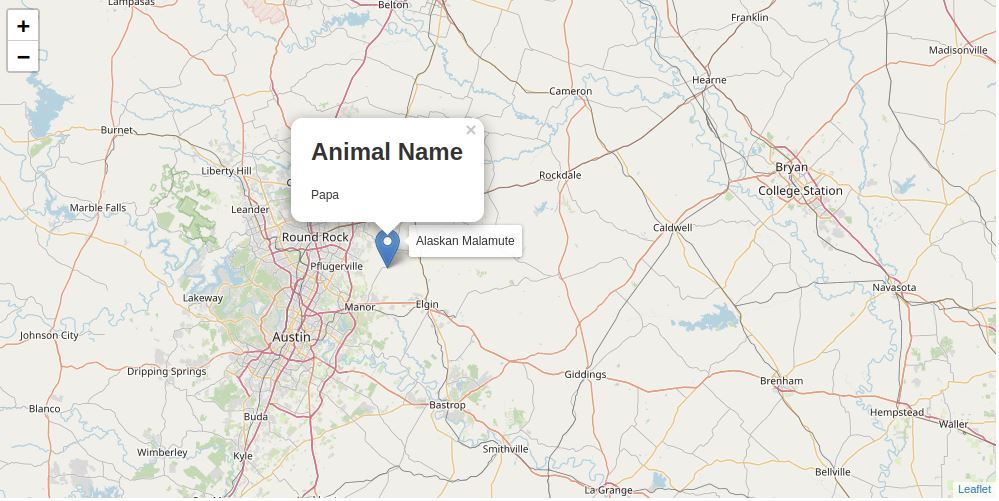
Description automatically generated

Disaster Rescue filter:



Pie Chart: The pie chart located in the bottom left of the page shows percentage of the breeds according to the present data. Currently the initial pie chart does not limit the minimum percentage, so the initial pie chart is not the best thing to look at.

Map: The map in the bottom right of the page shows the location of the selected animal, the first one by default. The map has a “pin” showing the location which can be hovered over to show the breed and clicked on to show the animals name.



Delete method: takes a query to find the element as an argument and returns the amount of elements that were deleted.



### Tests

The tests I have ran were in jupyter. Start by importing the library and instantiating the class. Then make your dataset an if statement will tell you whether the create method has inserted the data into the database or not. To test the read method, search for something in the database like what was just created, it will return a list of everything that matches that criterion, in this case just the single item that had been previously created in the example below.

A white background with red and green text

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

A close up of a text

Description automatically generated

### Tools:

## Contact

Your name: Brock Meek