# CS 340 README V2

## About the Project/Project Title

This is an app which utilizes Python to connect to a server and perform CRUD operations in MongoDB.

## Motivation

The purpose of the module is to simplify the process of connecting to a server and performing CRUD operations in MongoDB.

## Getting Started

To import the module, the animal\_shelter.py file must be placed in the same folder as the python file to be run. Following this, use the command “import animal\_shelter” at the top of the file, with the rest of your imports.

## Installation

MongoDB needs to be installed on running on the server which the module will connect to. Plus, the module requires the module MongoClient from pymongo and the module ObjectId from bson.objectid. Accordingly the package pymongo will need to be installed, on Ubuntu systems this can be done with the pip install pymongo command.

## Usage

### Code Example

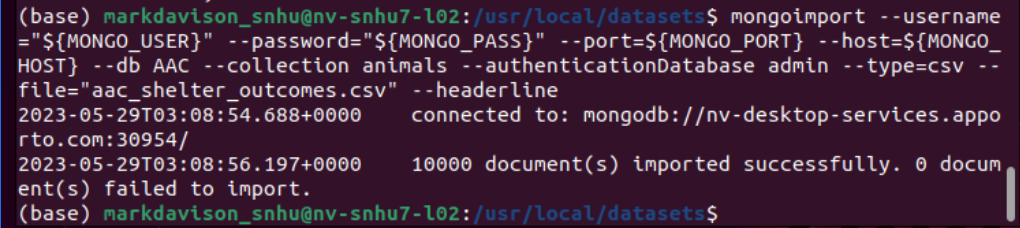


Figure 1. This image shows the importation of the database from a .csv file.



Figure 2: This image shows user authentication into the created database and collection

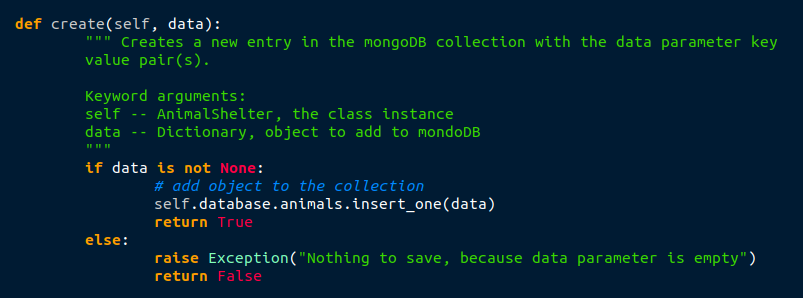


Figure 3: The create() method.

The create() method takes a dictionary parameter and the implicit parameter of the instance of the AnimalShelter class. Then it ensures the dictionary is non-empty and adds it to the collection of the database.

### Tests

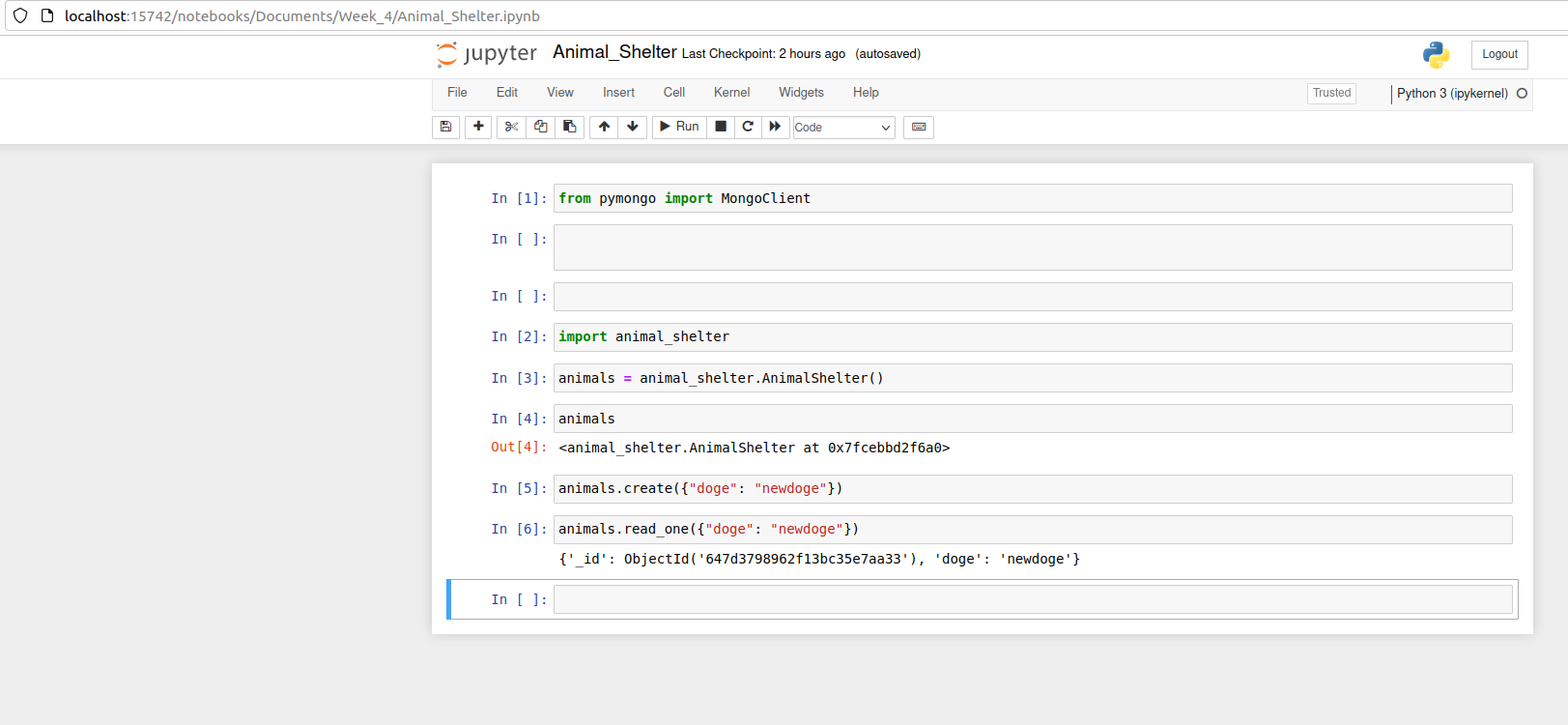
**

Figure 4: The create() method in action

In this example, the module is imported into a Jupyter notebook. Then an instance of the AnimalShelter class is created, which connects to and logs into the MongoDB server. Finally, a new entry is added using the create() method and then its details printed with the read\_one() method.

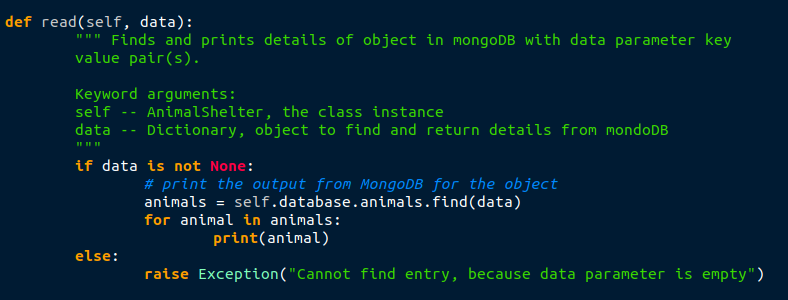


Figure 5: The read() method

The read() method takes a dictionary parameter and the implicit parameter of the instance of the AnimalShelter class. It then ensures the dictionary parameter is non-empty and then gets and prints the database entry for that dictionary.

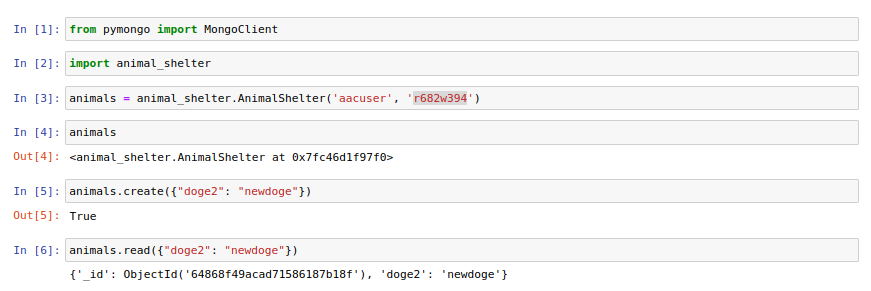


Figure 6: The read() method in action.

Here the module is imported into a Jupyter notebook. Then an instance of the AnimalShelter class is created, which connects to and logs into the MongoDB server. A new entry is added using the create() method and then its details printed with the read\_one() method. Lastly, the read() method is called to retrieve the information about the added object.

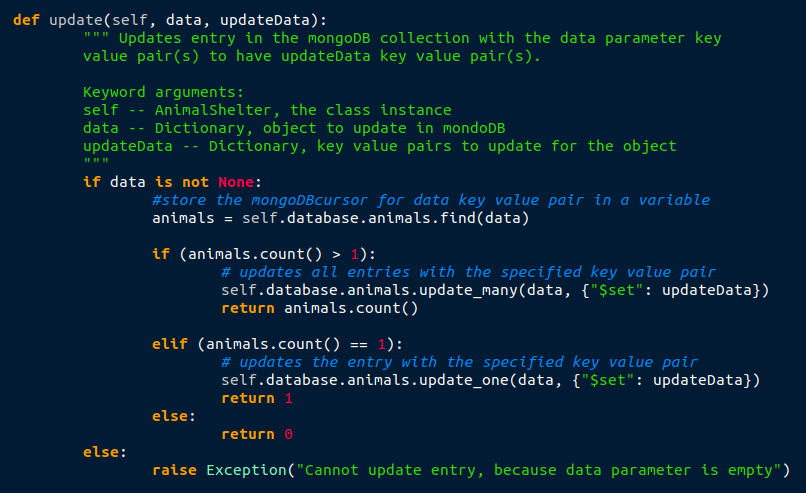


Figure 7: The update() method.

The update() method takes the dictionary parameters data and updateData as well as the implicit parameter of the instance of the AnimalShelter class. It checks that the data parameter is not empty then retrieves the mongoDB cursor of entries matching the data parameter. It then checks if the number of entries in that cursor is greater than one, and if it is it updates the fields listed in the updateMany parameter to the values in that parameter using the updatemany() method. if the number of entries in the mongoDB cursor is one, the same operation is done using the update\_one() method.



Figure 8. The update() method in action.

Here the module is imported into a Jupyter notebook. Then an instance of the AnimalShelter class is created, which connects to and logs into the MongoDB server. A new entry is added using the create() method and then its details printed with the read\_one() method. The update() method is called to add a new key value pair to the entries matching the newly added entry’s key value pair. Lastly, the read() method is called to retrieve the information about the updated object.

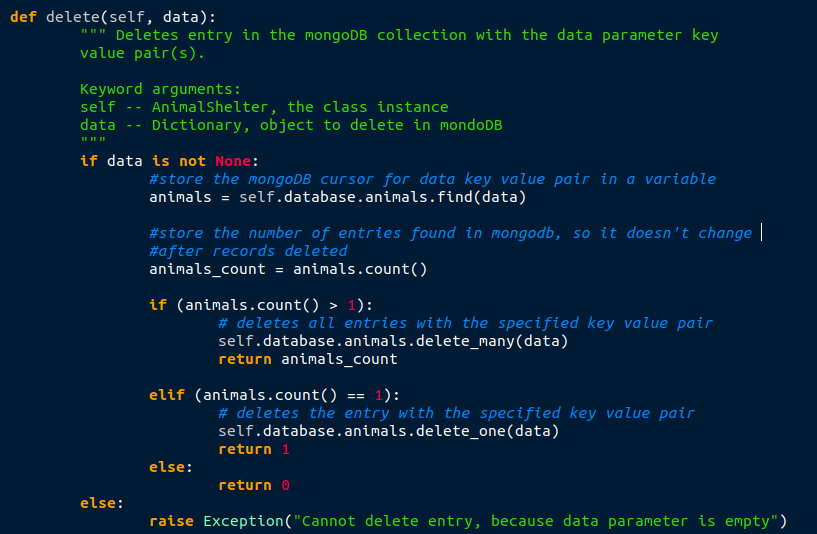


Figure 9. The delete() method.

The delete() method takes a dictionary parameter and the implicit parameter of the instance of the AnimalShelter class. It checks that the data parameter is not empty then retrieves the mongoDB cursor of entries matching the data parameter. If the number of entries in the mongoDB cursor is greater than one, it deletes all matching entries using the delete\_many command. If it only has one entry, the delete\_one command is used.

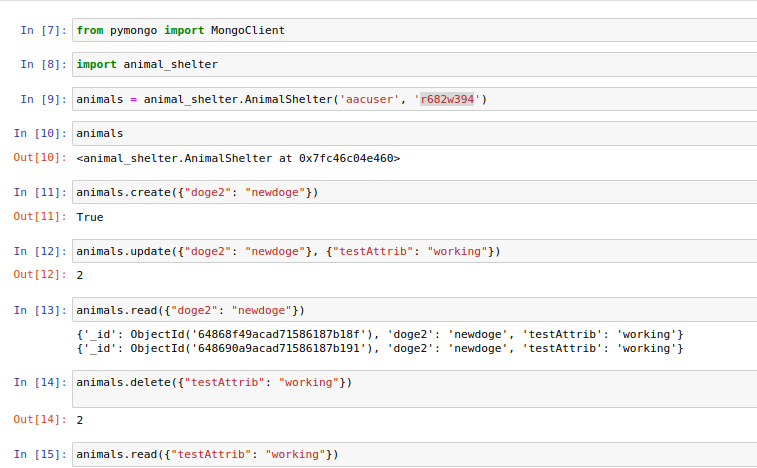
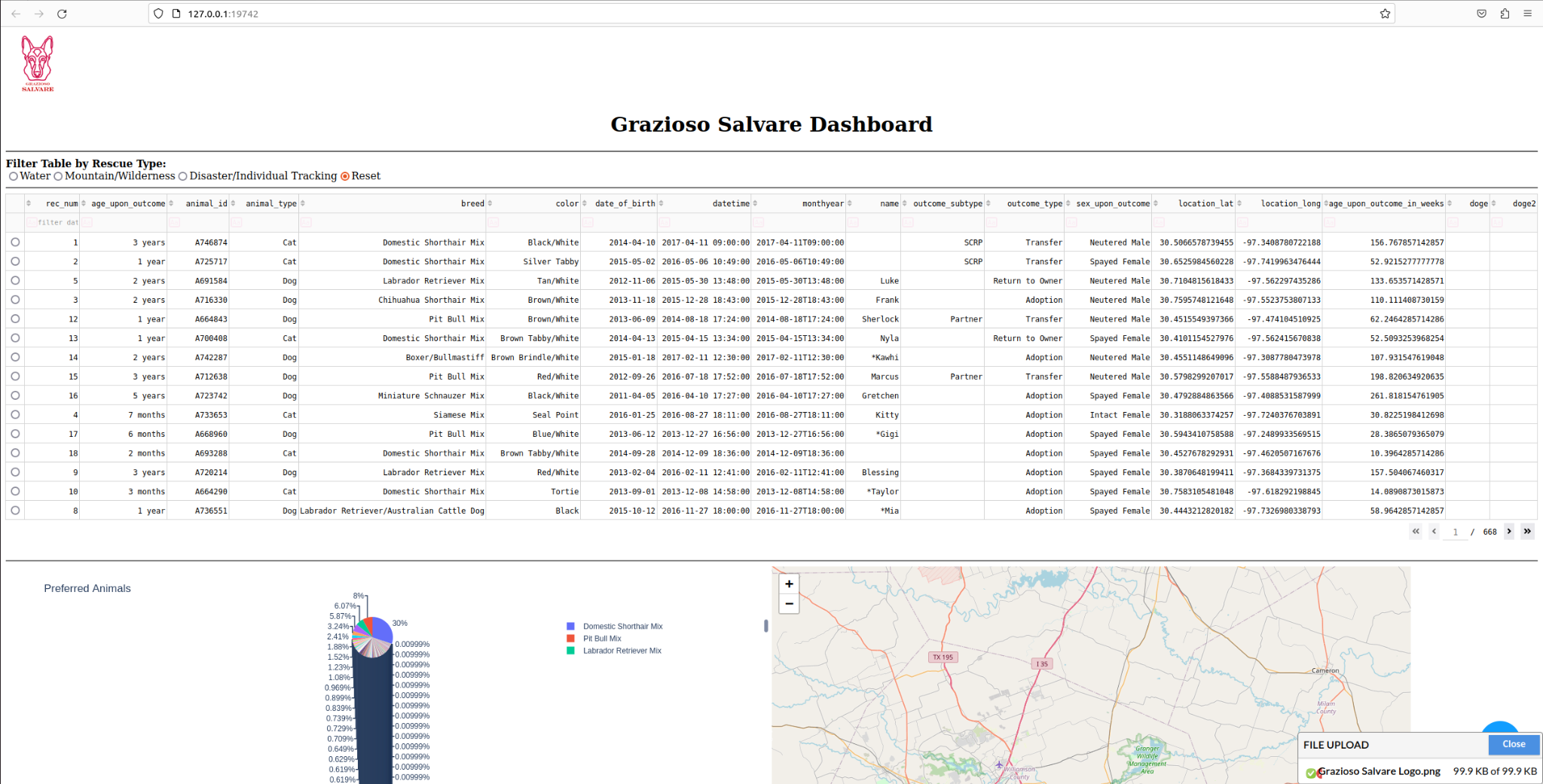


Figure 10. The delete() method in action.

Here the module is imported into a Jupyter notebook. Then an instance of the AnimalShelter class is created, which connects to and logs into the MongoDB server. A new entry is added using the create() method and then its details printed with the read\_one() method. The update() method is called to add a new key value pair to the entries matching the newly added entry’s key value pair. The read() method is called to retrieve the information about the updated object. Then all objects with the updated parameter are deleted, and the read() method is called to confirm their deletion.



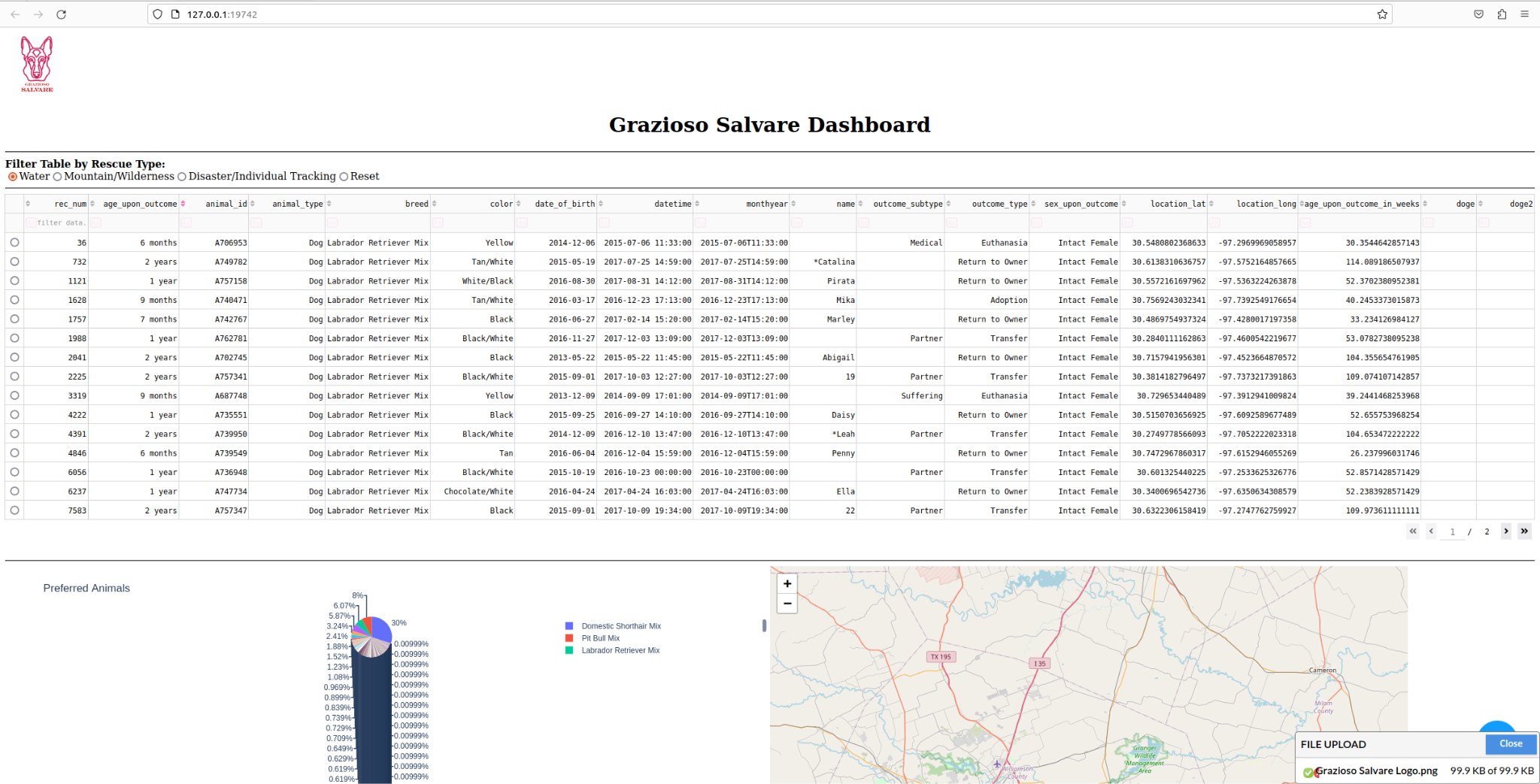


Figure 11. The effect of the radio button filters for search dog types

## Roadmap/Features

In the future we would like to add more advanced and detailed error handling, as at present the only handled error is if an empty dictionary is passed as a parameter.

## Contact

Your name: Mark Davison