# CS 340 README

## About the Project/Project Title

The project is about being able to apply the CRUD method of database operation using MongoDB. To do this, this project demonstrates the ability to use MongoDB integrated with python in order to create, read, update, and delete a document from the database, while implementing authorization.

## Motivation

The reason this project exists is to fulfill the goal of the client, Grazioso Salvare, to have a database that can be interacted with through a client-facing web application. This database will provide data across its many locations in Texas and this development will help identify and categorize their animals.

## Getting Started

In order to get a local copy of the application running, follow the following steps:

Ensure you import the database with the following command:

mongoimport --port ##### --db AAC --collection animals --type=csv --headerline ./aac\_shelter\_outcomes.csv (You can get the port number by running, /usr/local/bin/mongod\_ctl status, in terminal)

Add an Administrative user with the username of “aacuser” by following the guide located here: https://docs.mongodb.com/manual/tutorial/enable-authentication/

Download the file ModuleFourPython.py

Download the file ModuleFourScript.ipynb

Store these files in the same directory for convenience

Open the ModuleFourScript.ipynb file with Jupyter Notebook

Change the password at crudAnimal.\_init\_(“aacuser”,”password”), if needed.

Run the ModuleFourScript.ipynb and the data will be created, the cursor location will be displayed, the animal\_id will be updated and displayed, and the script will perform a deletion and show the results

## Installation

Python will be necessary to run the scripts for this project. Anaconda will be needed in order to access Jupyter Notebook. Jupyter Notebook can be run and installed with Anaconda, but it is not recommended. Jupyter Notebook will allow the user to test and run the commands without having to directly utilize the terminal. MongoDB is the backbone of this project and is necessary to achieve the results of this project with a database.

Python, can be installed in terminal with this command: sudo apt-get install python3.6

Anaconda, can be installed a here: https://www.anaconda.com/products/individual#linux

Jupyter Notebook, can be with command: conda install -c conda-forge jupyterlab

MongoDB, can be installed by following these instructions, https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/

## Usage

In module three of this assignment, it was necessary to import the database for the animal shelter using the import commands included above. Additionally, adding user authentication was done in order to add a layer of protection and authorization to the database. This was done using the instructions above, which takes important steps to add database administrator level users.

The create and read functions were made by specifically, ensuring that the database was included in the function call, in order to allow the code to be re-used across database changes. Furthermore, the username and password were imported to initialize the database. The create function made sure to import the json data that was loaded, and the read function incorporated the animal\_id that was made in the script. The code also makes sure to use if else statements in order to throw errors or allow continuation, respectively.

The update and delete functions were able to be made much easier in comparison to the prior two functions. The update function was designed to not only update the value, but to display the old value for comparison and confirmation. The delete function was very straightforward but ensured that the user was able to see the number of items deleted and their status. These functions used OOP and were error handling, combined with proper coding practices were exercised.

### Code Example

*from ModuleFourPython import AnimalShelter*

*crudAnimal = AnimalShelter()*

*animal\_id = 'A'.join(random.choices(string.ascii\_uppercase + string.digits, k = 6)) #creates the animal\_id*

*randomDigits = '1'.join(random.choices(string.ascii\_uppercase + string.digits, k = 5))#creates the id*

*dataSet = … #the dataset would be here, removed to be concise*

*crudAnimal.\_init\_("aacuser","password")#access to mongo with authorization*

*crudAnimal.create(dataSet, "AAC")#creates the animal*

*dataRecord = crudAnimal.read(animal\_id, "AAC")#reads the animal and stores in variable*

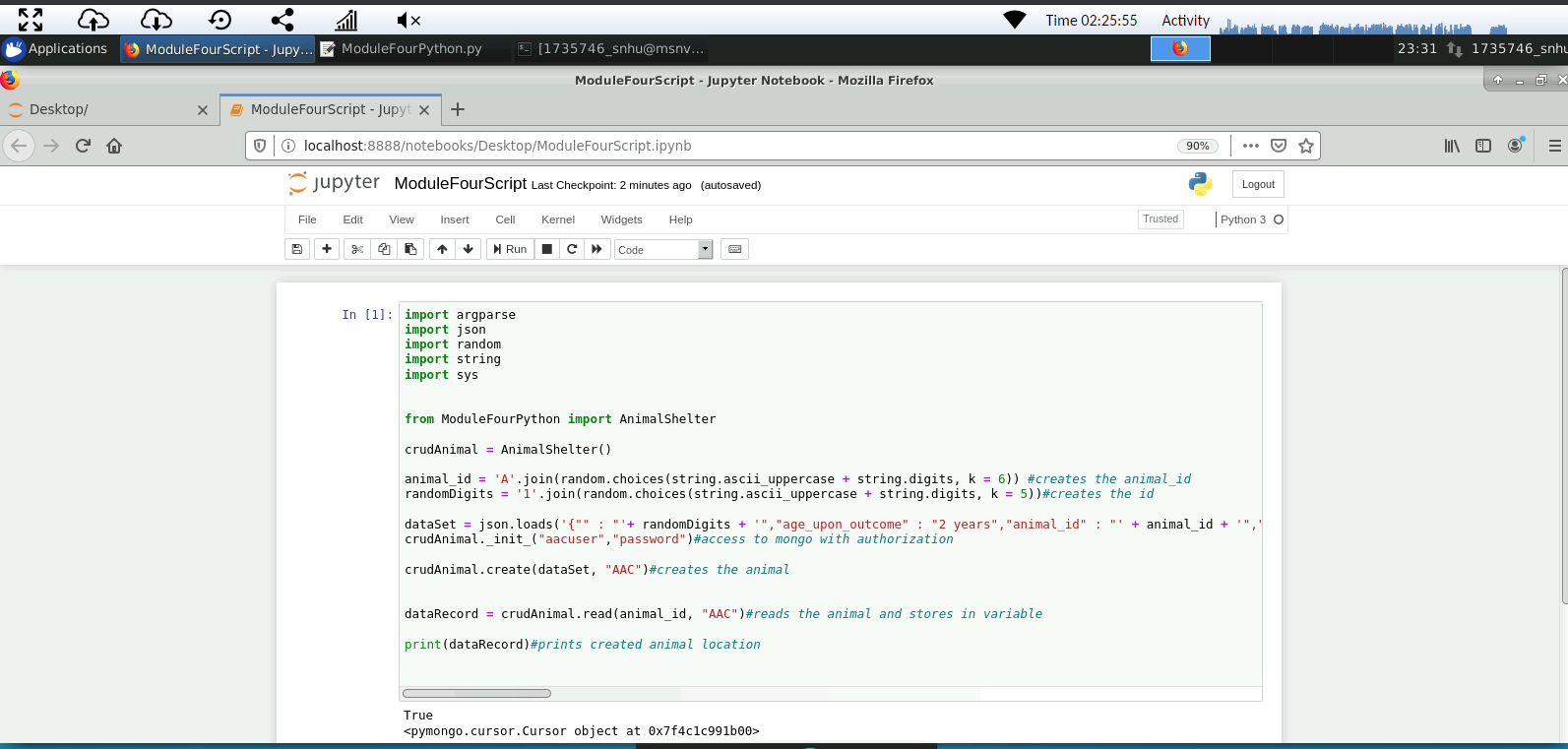
*print(dataRecord)#prints created animal location*

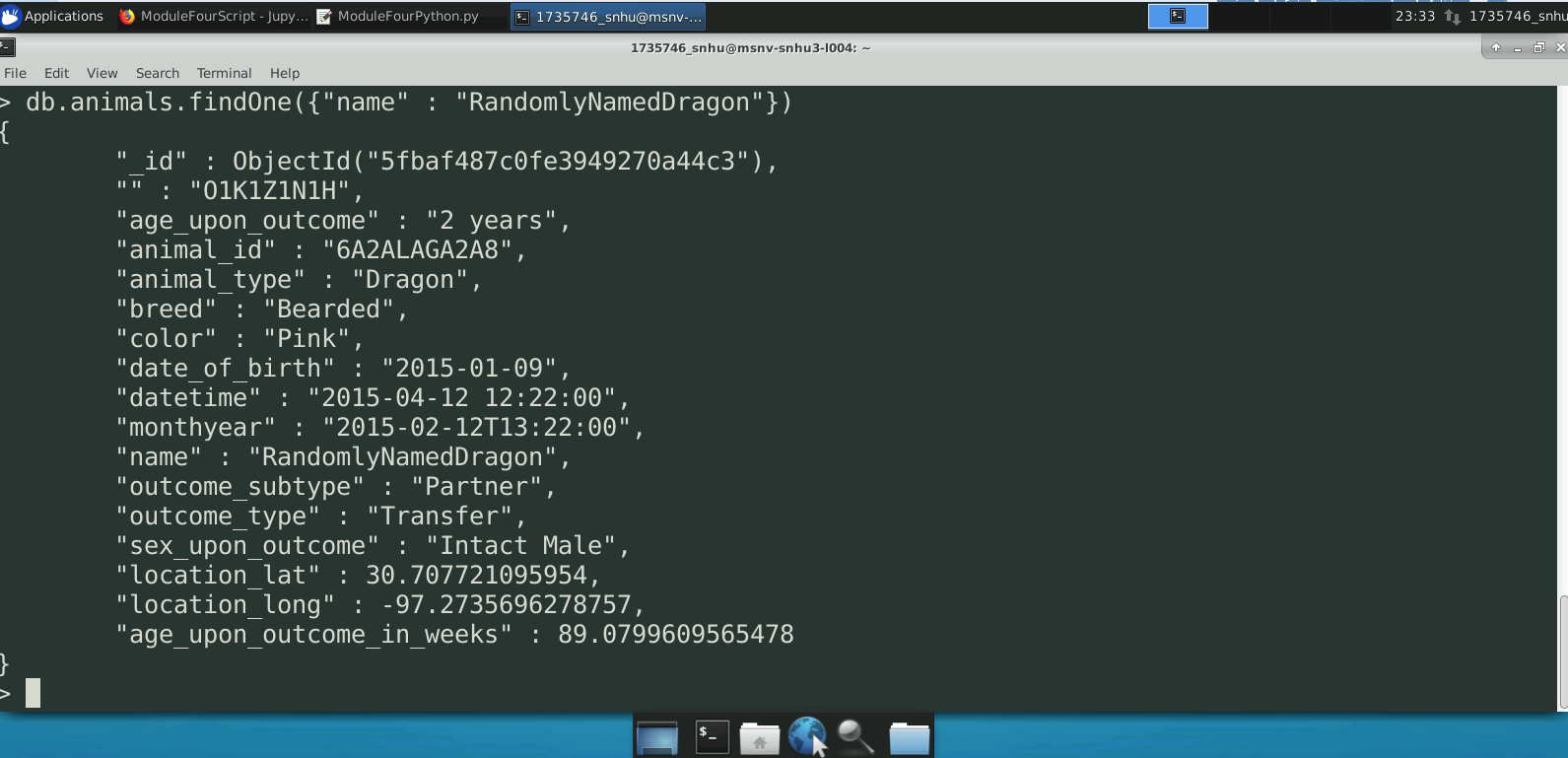
The code above is from the .ipynb script which executes the ModuleFourPython.py file while allowing for code reusability. As you can see above, the animal\_id and random\_digits are created by random. Following this, the dataSet is generated. Then, mongo is initialized, and the dataset and database names are sent to create the animal. Following this, the animal\_id and database name is sent to read the animal, which has its cursor location printed.

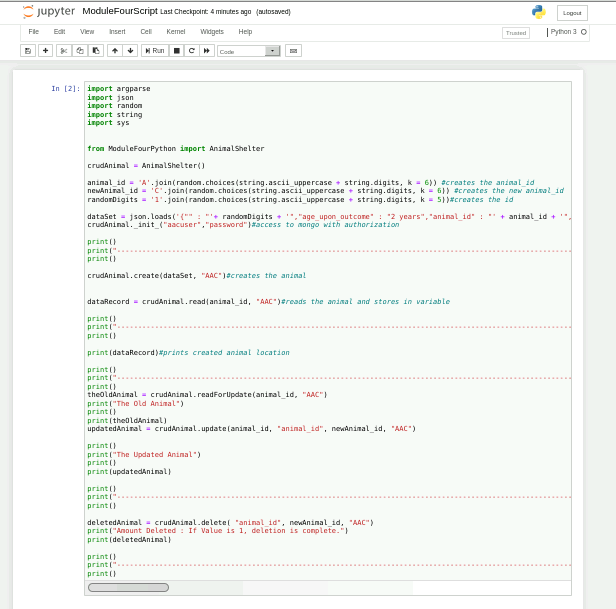
### Tests

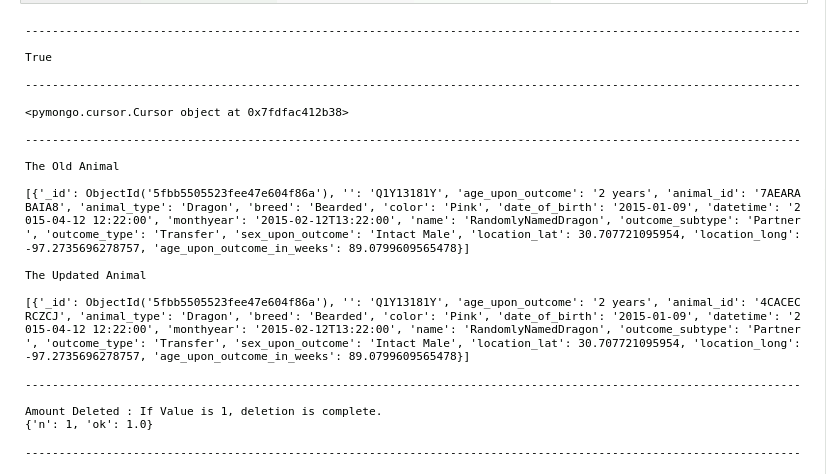
In order to successfully test individual portions of the code through testing, this can be achieved by breaking the code in the .ipynb file into code blocks via Jupyter Notebook. Another way of testing the functionality is to run the code and put ‘test strings’ in each section. For example, the main reason that the randomDigits variable was added alongside the random animal\_id was to ensure that there were not duplicates of these values, to aid in testing. Then, between sections such as when the create and read functionality is ran, we can add strings such as, print(“This code ran successfully”), and delete it before release. This was done during the writing of the code to make sure each section was operating and was removed for the neatness and portability of the code.

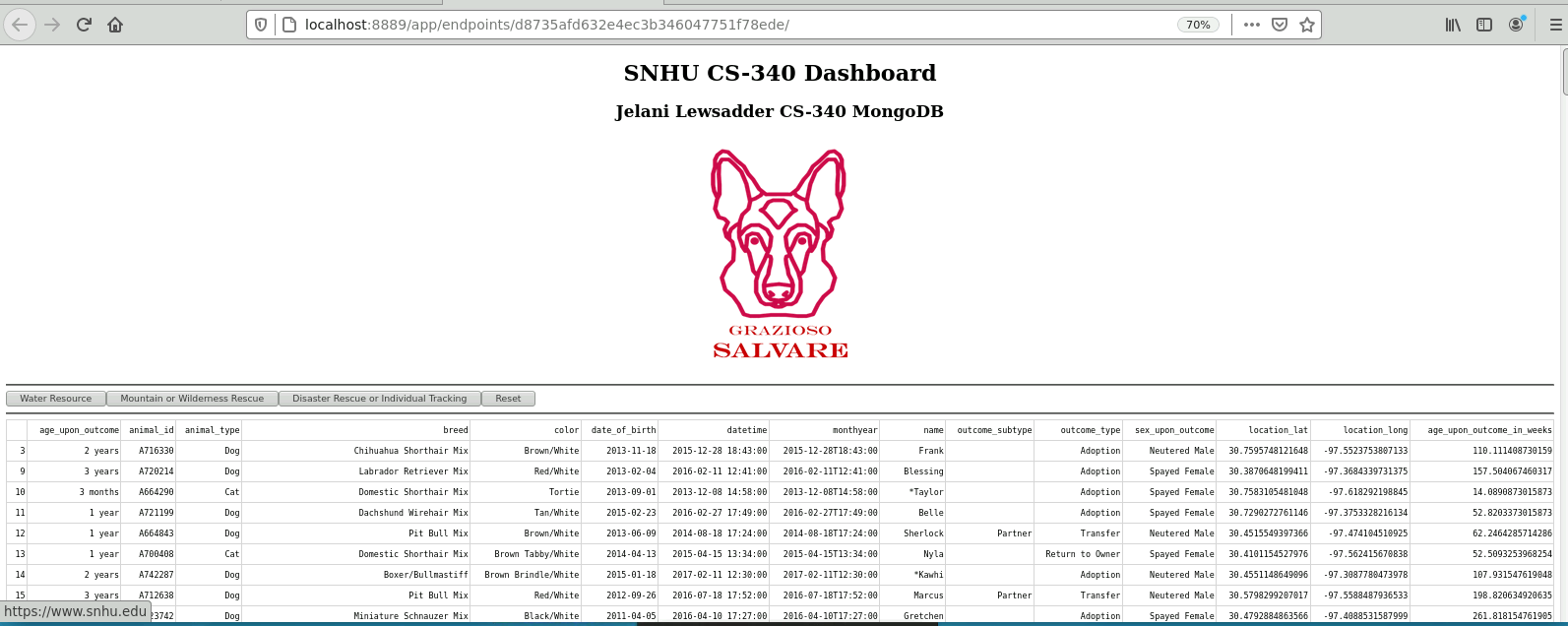
### Screenshots

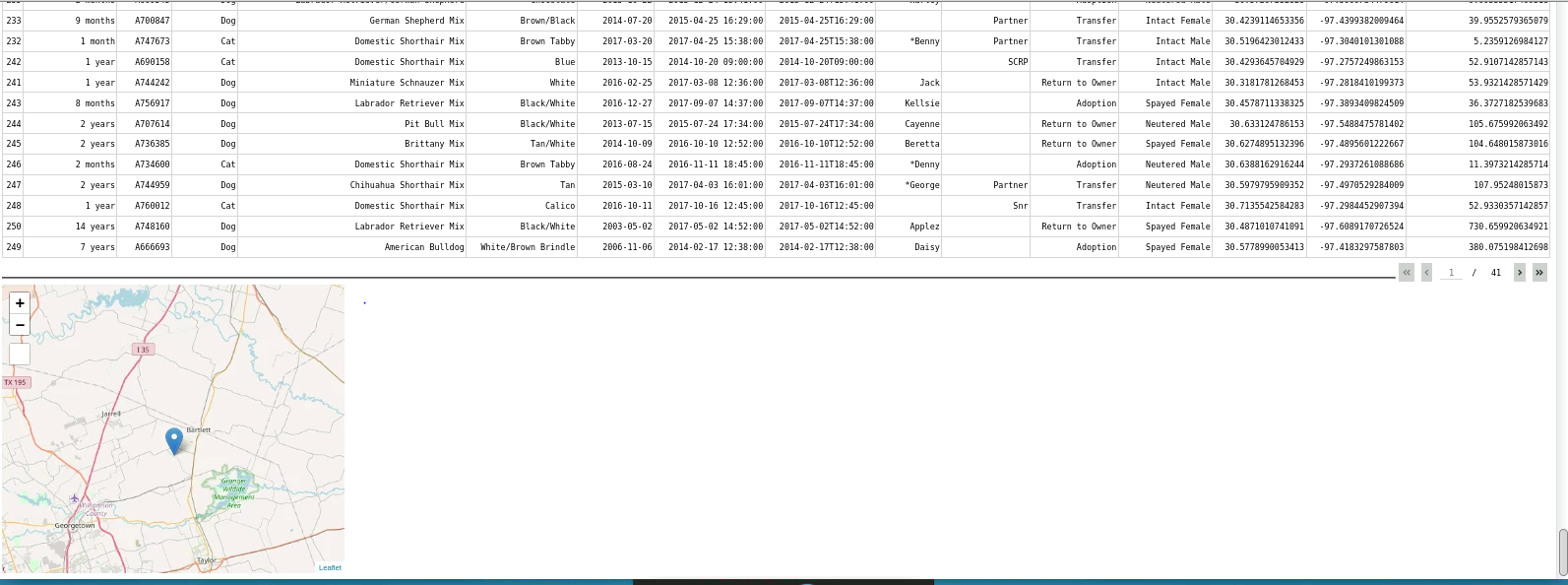


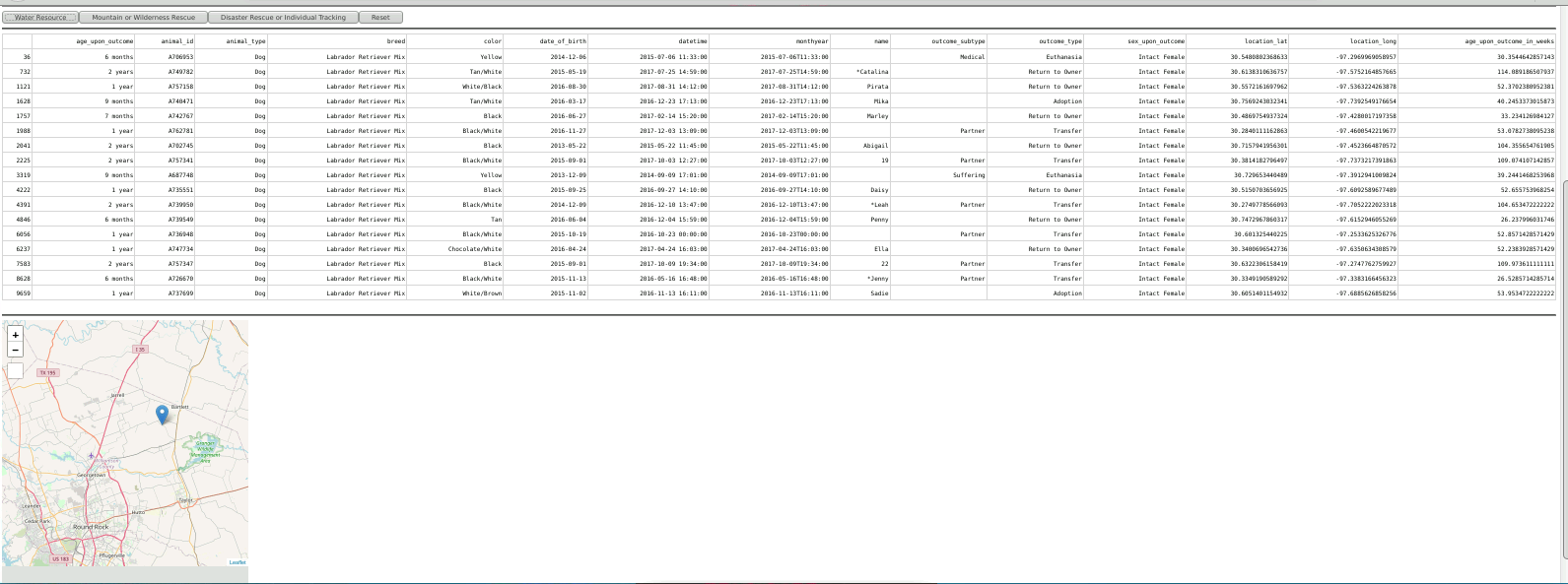


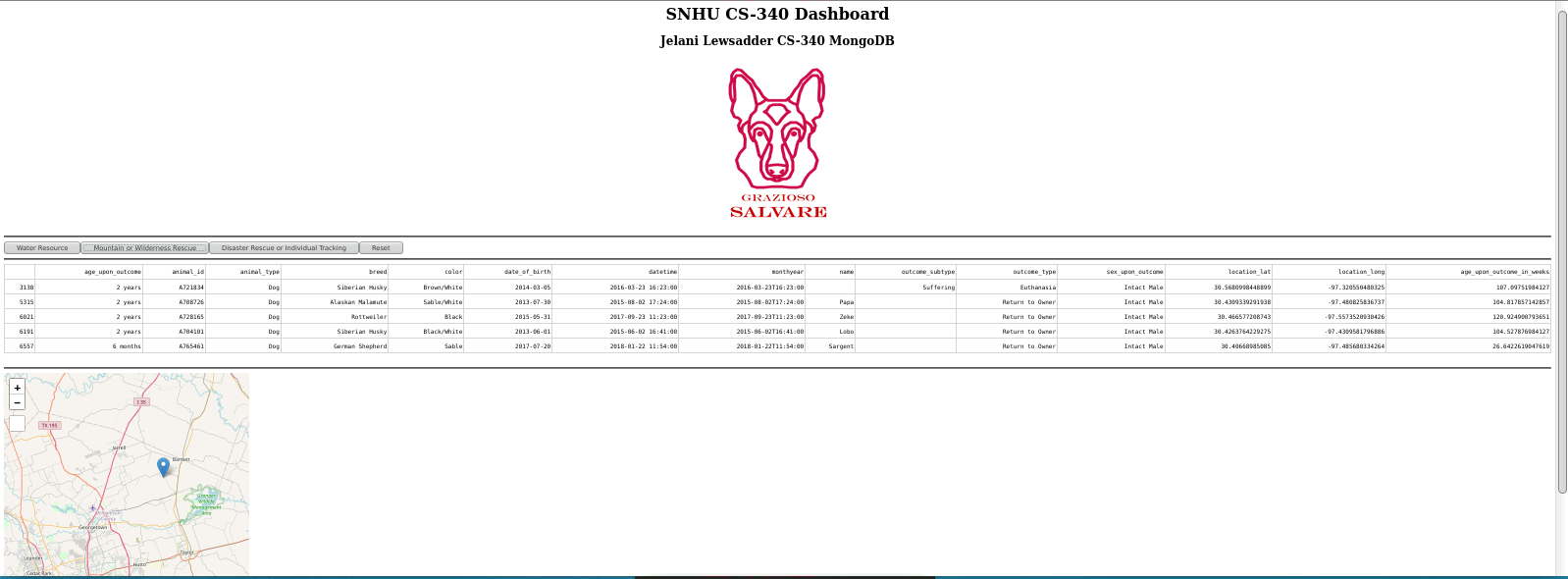


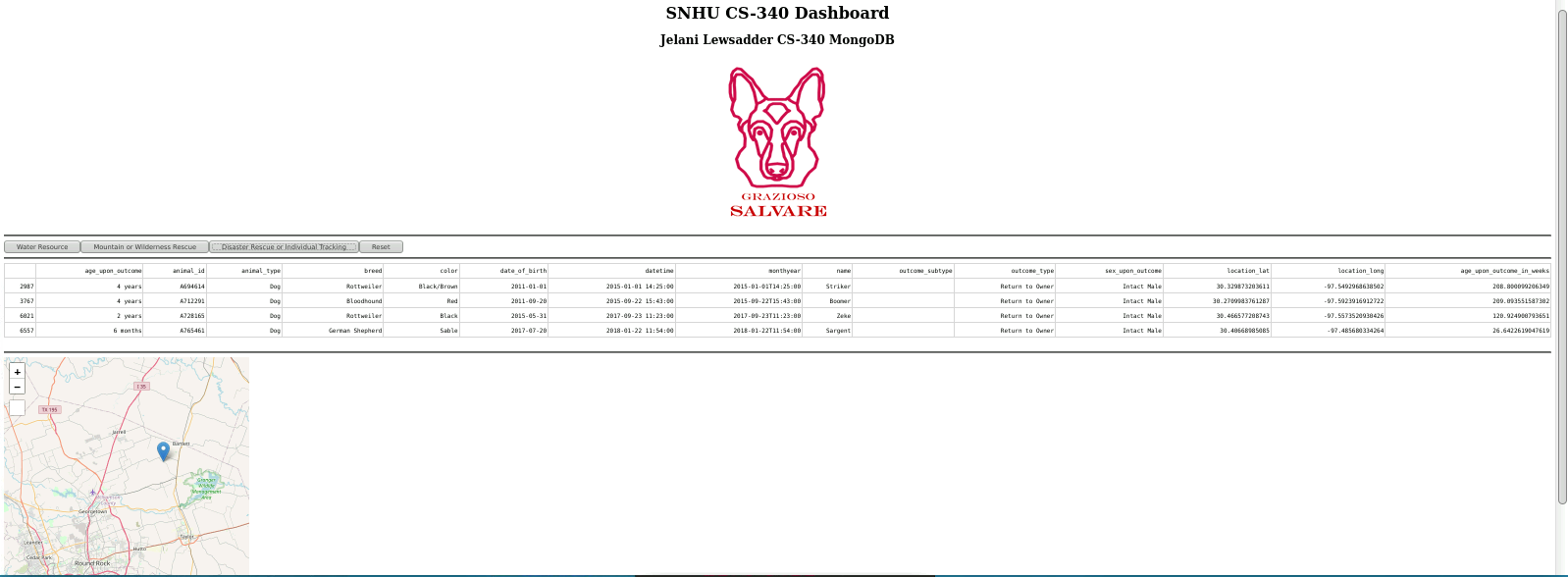


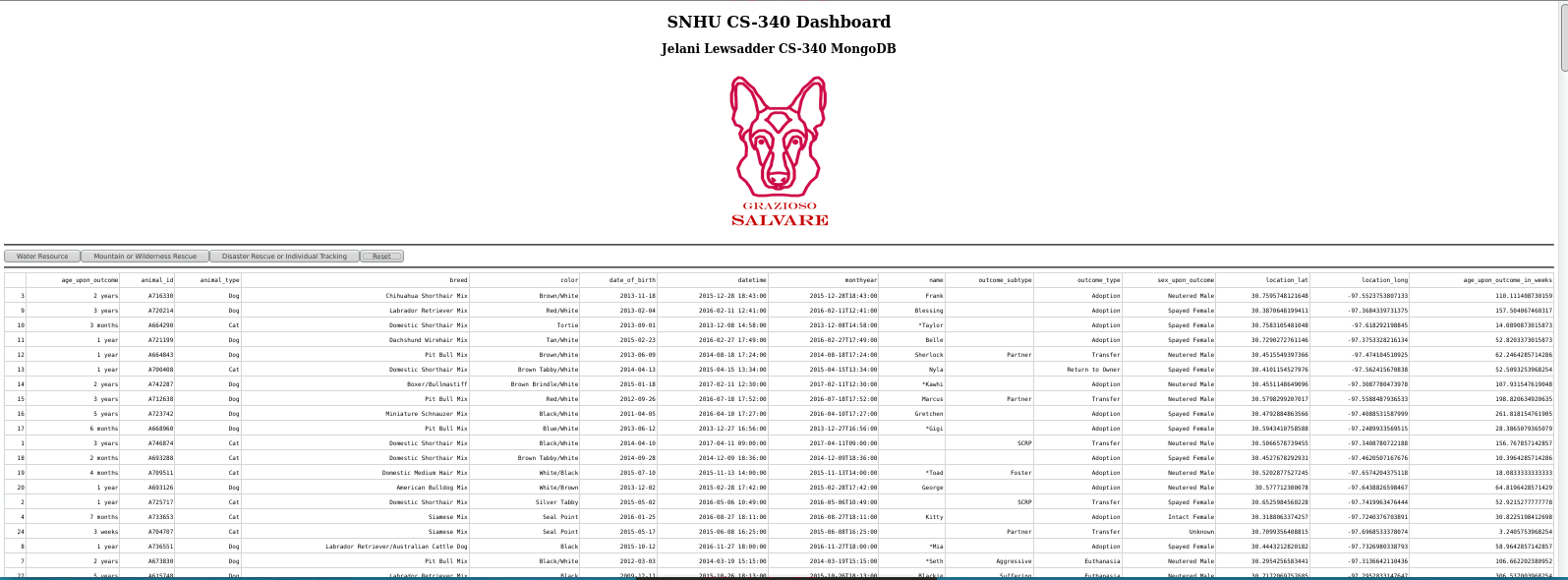














**Purpose of The CRUD Python Module**

The CRUD python module has a very high level purpose in that it allows the Python Script to operate with the code that needs to be accessed in order to achieve the goals of CRUD(create, read, update, and delete). This module being available also allows the code to be modular and reusable, considering that in the future only the Script needs to be edited.

**How The Module Should Be Used**

The python driver that was used in order to accomplish the task was the PyMongo driver. It allowed many of the needed features such as the updating and deleting values but also displaying them in Json with bson.json\_util import dumps and bson.json\_util import loads. This was necessary in order to achieve the result of using JSON and it ended up working very well for the project.

The CRUD operations were created with allowing them to be concise and reusable. With this in mind, the value of data being input can be changed in the script; specifically, with the update and delete operations. They are not limited to just then animal\_id. The modules allow for data to be created and allow the user to see if the data was in fact made with proper print statements and error handling. The read operation searches for a value and gives the cursor location to the user. The update and delete operations use properly built parameters and naming conventions which allow for flexibility, error handling, and proper print statements for the user to identify the results of the operation.

**Describe the required functionality**

It was required that the project have the name of the designer, the logo of the company that links to the SNHU website, a data dashboard and filtering options, and widgets to interact with the data dynamically. The Screenshots are in the Screenshot section above.

**Describe the tools used to achieve this functionality and a rationale for why these tools were used.**

MongoDB was used as the modeling component for this application due to the ability to use documents and python to query files and apply data visualization. The ability to work fluidly with python and dash components is a big plus to its usage. The Dash framework provides the ability to use html resources with python in order to create a separate web page that can interact with the database on the backend through Linux. This is a very powerful tool and a big reason that these tools were able to be used synonymously.

<https://plotly.com/python-api-reference/generated/plotly.express.pie>

<https://dash-leaflet.herokuapp.com/>

https://dash.plotly.com/dash-core-components

**Steps that were taken to complete the project.**

In order to complete this project, it was important to create and test the different components of the application. The first step was to call the CRUD module and to add the image with the link the snhu website. Next, the buttons were added to filter the data, along with their logic to operate on click. After this, the filter options were added and linked to the buttons. Then, the map widget was added and configured, and the application was tested and refined to ensure that the components were error free and the design of the dashboard was optimal.

**Identify any challenges that were encountered and explain how those challenges were overcome.**

The main challenges that were needed to overcome were to design a nice-looking dashboard, incorporating proper queries, and making the button clicks work by only having to click them once to apply the proper filters. The dashboard was made to have a quality look by scaling the size of the image down and moving the headers to the top of the dashboard. The image was placed below these.

The queries challenge was resolved by very carefully creating the first query and testing it within the dashboard and with the buttons. The query took a good amount of time to build and a challenge with these queries was to realize that ‘$’ arguments must be wrapped with quotations to work with python.

Finally, the button clicks only having to be clicked once to filter instead of increasingly based on a more than variable is one that took a bit of creativeness to correct. At first, the solution was to try and measure the variable size within the function and to reset number of clicks to zero, but these did not seem to be effective or efficient. Instead, global button variables were created that would increase by 1 every time a button was clicked and filtered with. As long as the number of clicks was greater than the associated global, the filter was applied.

## Contact

Name: Lewsadder, Jelani