# Project Two README

## About the Project/Project Title

*This will allow for the Grazioso Salvare project to interface with the database in a much more user-friendly way and allow for them to avoid learning MongoDB commands, and instead to work with the Python controller. The project will allow for the company to easily search for specific dogs through the act of filtering by words, or even more importantly, they are able to use the preset filters to search for specific types of working dogs. This means they can search for dogs that fit all kinds of rescue animal requirements.*

## Motivation

*The motivation behind creating this project was to provide the international rescue-animal training company, Grazioso Salvare a way to easily identify dogs that are potential matches for their training.*

***To get a local copy up and running you need to complete the following steps:***

1. *Install or Have MongoDB installed with a Database Setup*
2. *Import the CRUDS.py and ProjectTwo.ipynb files into Jupyter Notebook*
3. *Launch your local MongoDB with authentication disabled*
   1. *Create a User Account*
   2. *Exit MongoDB*
4. *Enable authentication and restart MongoDB with the command below* 
   1. */usr/local/bin/mongod\_ctl start*
5. *Modify the code to connect with your local MongoDB and to connect to your user account*
6. *Once you have done that, you are ready to run your ProjectTwo.ipynb file and see the final dashboard. From here you can view the tables, chart, and map. You can even see how they change as you filter or move through pages.*

***Authentication:***

*During week three we set up admin and aacuser access to the database. Admin has control over all of the databases and aacuser has access and privileges on only the AAC database that we are using for this project. We used aacuser to log in to the database to allow us to modify the contents and search the contents of the AAC database.*

*Pymongo provides us with a great deal of functionality when interacting with MongoDB. this means that after we verify that the input isn’t empty, we can insert it into the database with the insert\_one function if we want to insert a single document as we did here. With the Read portion, we wanted to use find to search for documents based on the key-word pairs available.*

Installation

*List the tools you need to use the software and how to install them.*

* *MongoDB*
  + *MongoDB was a great choice for this project. There is a great deal of functionality that we are able to access with Python. For a variety of reasons, MongoDB is a great tool, but when we combine it with Pymongo, we reach a level where we are able to automate and simplify our interface with the database. MongoDB, in my opinion, is also quite a user friendly compared to other databases I’ve worked with but the other databases were SQL databases so that may have been the difference.*
  + *Download From* [*https://docs.mongodb.com/manual/installation/*](https://docs.mongodb.com/manual/installation/)
  + *We utilize a variety of commands for this project.*
    - *createUser - We create the admin and aacuser accounts*
    - *find & findOne - Allows us to read from the database*
    - *insert - Allows us to insert new documents into the collection*
    - *update - Allows us to update a document in the database*
    - *delete - Allows us to delete a document in the database*
* *Jupyter Notebook*
  + *Download From jupyter.org*
* *pymongo*
  + *pymongo is imported into the CRUDS.py file*
  + *This is necessary to use at it imports functionality for us to connect to a MongoDB server and to interact with the databases.*
    - *Specifically, we use pymongo to create an instance of MongoClient which allows our object to connect to the MongoDB as you can see from the screenshot below.*

**

* + - *After we instantiate the object, we can use commands to interact directly with the database through that object.*
* *Plotly*
  + *I used Plotly to construct my pie chart*
  + *Besides Plotly being easy to use, there are many benefits where I can see myself potentially using it in the future if I need to develop charts.*
    - *My favorite part is that you are able to download pictures of the charts that you create with Plotly right on the web application.*
* *Dash Framework*
  + *The Dash Framework is what provided us with the view and controller structure for the application.*
  + *From the research that I did about Dash before starting this project and the information I gained during the use of Dash for the project there seems to be one area where it shines. This is in the space of data visualization.*
  + *Dash allows us to create visuals while also helping us to place those visuals into the UI of the web application.*

## Steps to Complete Project

* The first step that I had to take was in the development of my CRUD module.
* After this step was completed, I was ready to take on the rest of this project which involved constructing our dashboard.
* I started this process by accessing the data and using the data to create our table.
* After I set up the UI part of the table, I moved onto the map.
* The map required me to modify the app layout to place it below where the table was placed in our application.
* After doing this I developed the code for the map and as you will read below in the challenges, it was a tough task.
* I was able to find a solution to storing the latitude and longitude values to allow for the map to update with the table.
* After this, I added in the Grazioso Salvare logo which was as simple as downloading the image and uploading it into the jupyter notebook.
* From here I needed to work on getting the new filtering menu working. We had some starter code that was quite useful, I created a div in the layout for the dropdown menu. I used a dash import dcc.Dropdown for my menu.
* After the front end for the menu was created, I went to the backend where I used if and elif statements to compare the user’s selection to the different filtering options to provide the correct data to the users. Essentially, I would take input, and based on the input, I would reload the data for the table, but with a new query that supports the filtered choice.
* After the filtering options were complete, I turned to the pie chart.
* For the pie chart, I again had to add a div into the layout but I actually had this completed as the chart and map needed to be placed together and that was part of the starter code. After the chart was in the layout, I needed to develop the chart so we would actually have a chart to view. For this, I converted the data to a dataframe, used value\_counts to create a new dataframe with the breeds of dogs and the number of times that breed appears on a given page. This was a decision that I had to think about because I struggled with finding the value\_counts function that made my life easier. As I worked through this problem, I had plenty of bad ideas, but I think that was a good one. Finally, we return the dcc.Graph which contains a variable called figure which we set equal to the pie chart. The function px.pie took three parameters, the dataframe, values, and names. “Names” is the breed column of our dataframe and “values” is the number of times a breed occurs on the page.
* That was the last step in completing the project, besides testing of course.

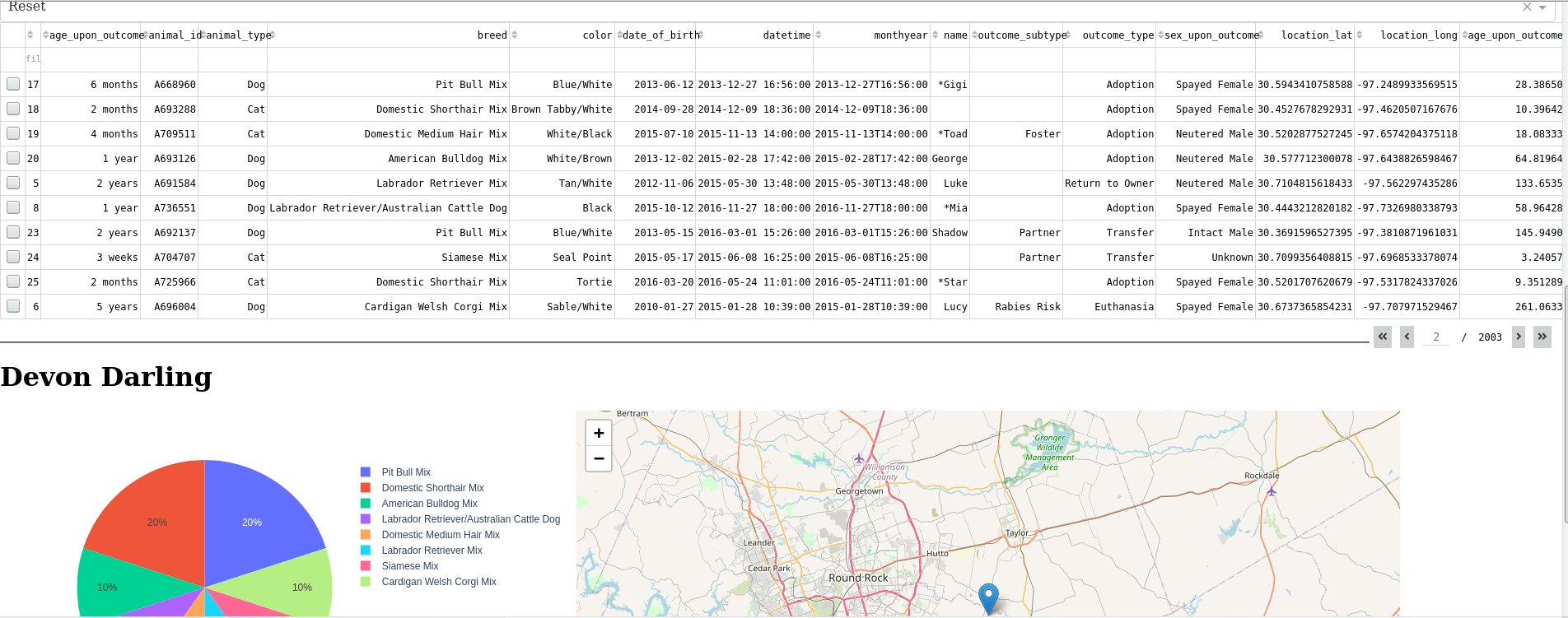
## Challenges

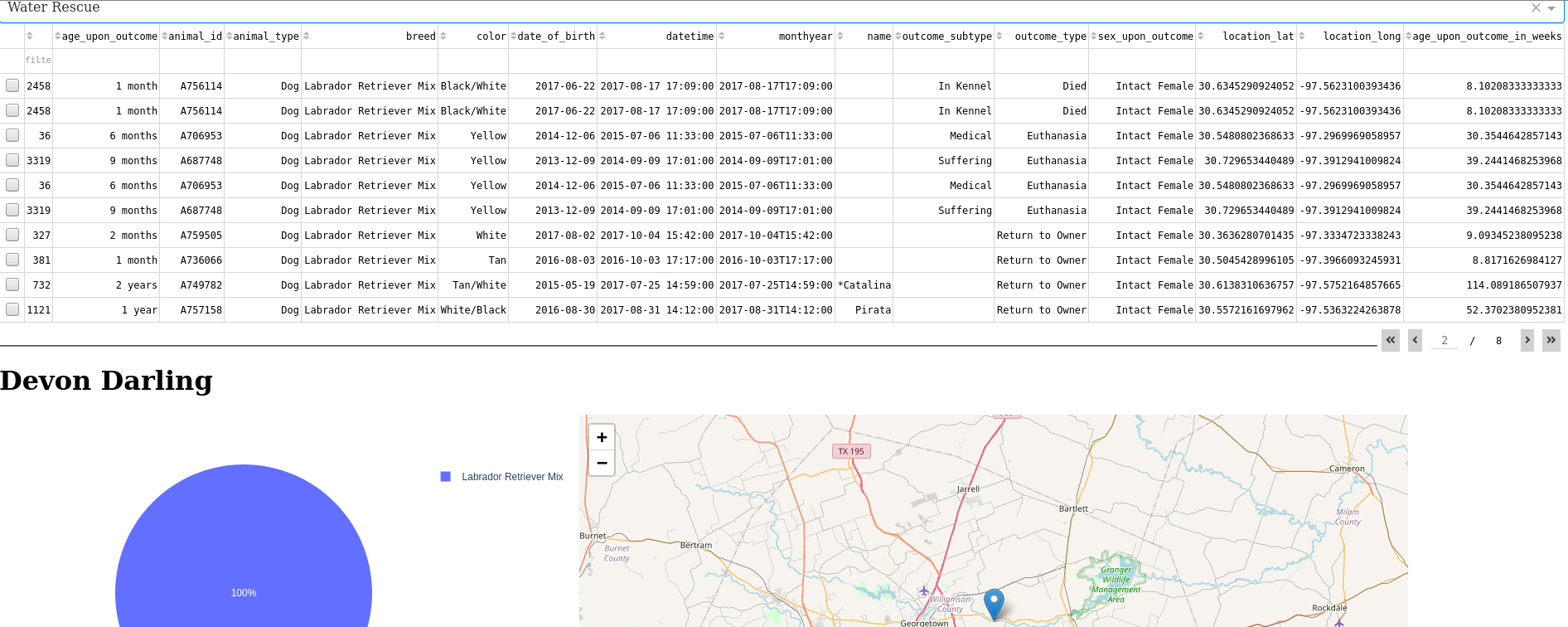
*A project is never simple when you start diving into it, but challenges are nothing to be concerned about. The only thing that matters is that we power through them to finish the product. I experienced a few challenges during this project. Some were complex while others were one-second fixes.*

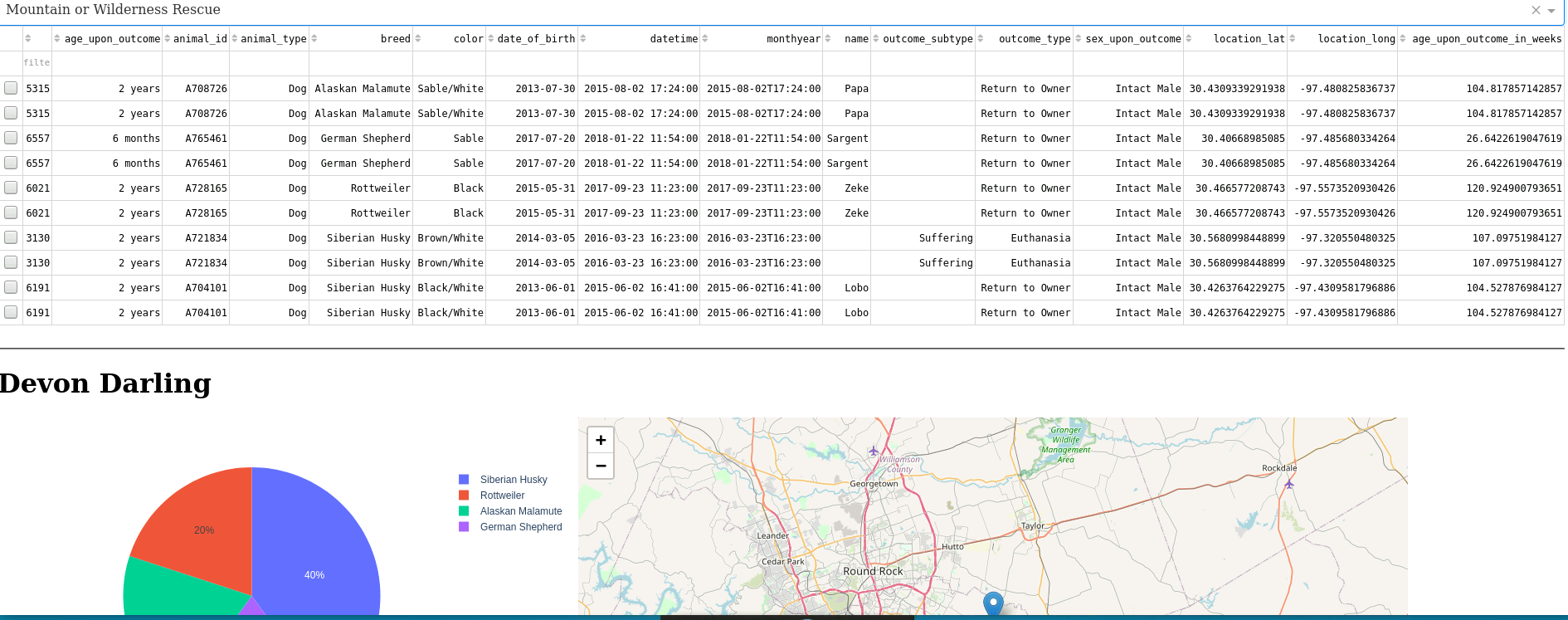
* *Getting the Drop-Down menu to return the value needed for the filtering function*
  + *I struggled with this because of the starter code that we received. That had the parameter of filter\_type but when I was learning about the drop-down menu, I found that you should use value to signify the value attached to the label you provided. After doing this, I struggled but eventually realized that changing the input parameter for the filter function would solve my problem and it did.*
* *Another challenge that I faced was having my map update.*
  + *This was more of last week’s challenge but it was also part of this project so I am including it here. The challenge was finding a solution of how I can pull data from the dataframe to have the map update because the starter code simply had hardcoded latitude and longitude coordinates. After a bit of research, I found a simple solution. Using “dff.loc[df.index[1], 'location\_lat']” for the latitude and “dff.loc[df.index[1], 'location\_long']” for the longitude, these functions allow us to pull data from the table and use that information to update our map. I stored these two pieces of code as variables to avoid needing to type out the entire thing multiple times or copy and paste it. I find creating variables is useful as long as they are easy for people to understand and I believe I accomplished that here.*
* *The last big challenge that I am going to talk about is my problems with the pie chart.*
  + *This was the most difficult part of the assignment in my opinion. I racked my brain for so long trying to find a solution to this problem. The starter code was a great help to point me in the right direction but I was lost. I started by thinking about iterating through the breed column to make a tally of the different breeds and how many times each occurs but that seemed pretty inefficient to me and I kept thinking there had to be another way. Eventually, I found myself in an article talking about the value\_counts() function which allows you to specify a column in your dataframe and it gathers the number of times each name is repeated in my case. Then I made two columns for this new dataframe that I created, the columns were “breed” and “value”, and after that, I was able to go into the return statement and construct my pie chart. The exact code was “figure = px.pie(new\_df,values="value",names="breed")”. This worked very well*

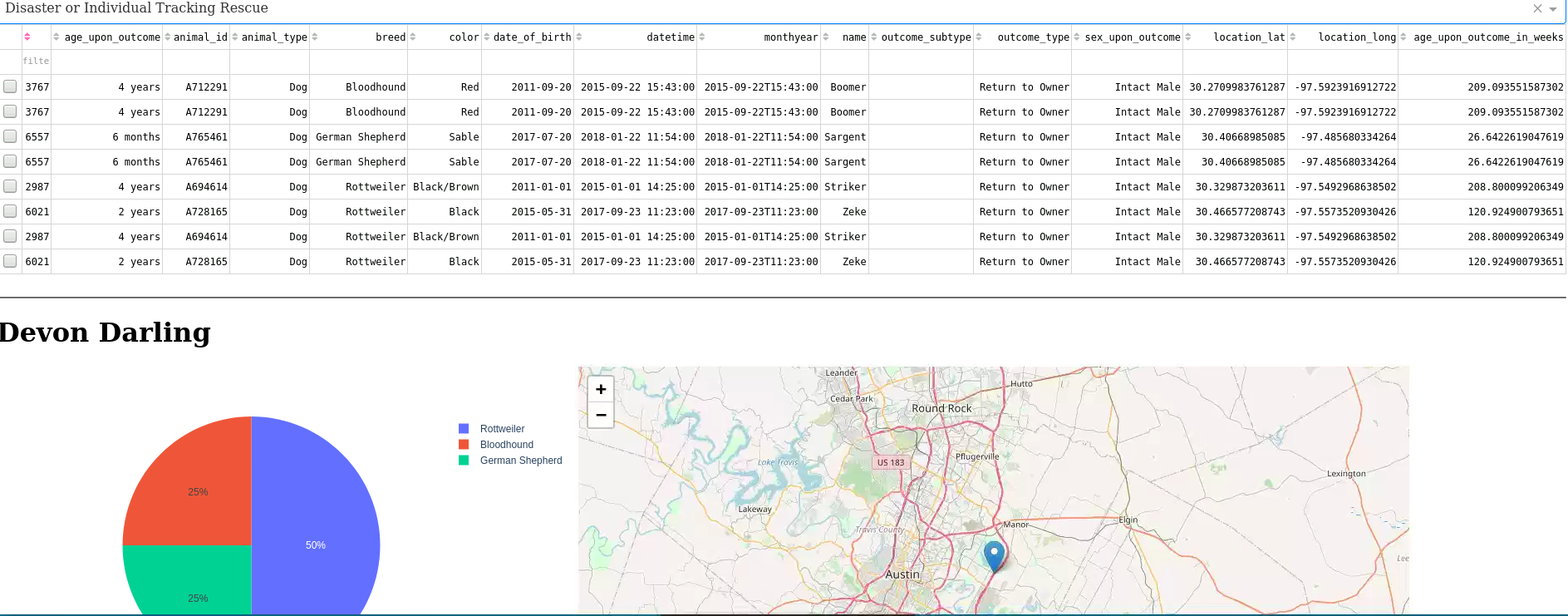
## Usage

*Unlike previous assignments, this one didn’t have software tests specifically but instead, functionality that we needed to create. Below you will see the four different filters that I implemented (Water Rescue, Wilderness Rescue, Disaster Rescue, and Reset). Reset is the initial state of the dropdown menu and is also used to reset the filtering of the database to show users the unfiltered data.*



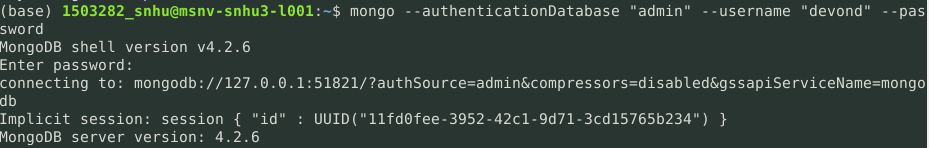




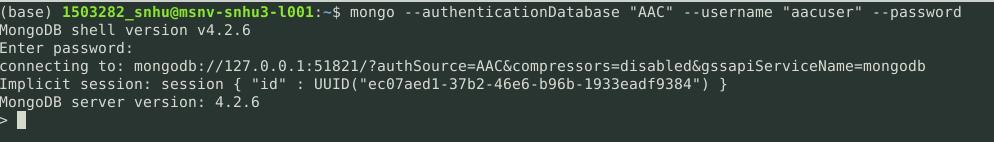


### Screenshots

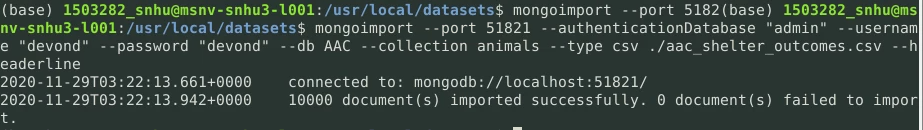
Administrator Login



aacuser Login



Import Documents



## Contact

Devon Darling

[devon.darling@snhu.edu](mailto:devon.darling@snhu.edu)