README

Global Rain

Grazioso Salvare Search and Rescue

**Project functionality:**

The Salvare animal tracker needed direct access to the AAC animal database hosted through Mongo. This information then needed to be presented in an easy-to-follow format for the user. This was achieved by presenting the information in a chart that separated each animal and presented all prudent information like age, sex, name, and breed.

It was also important that the user could interact with the chart of animals in some meaningful way to create a user-friendly application. This was achieved with a dropdown menu that allowed the user to narrow the dataset to specific types of rescue dogs. These types were Water Rescue, Mountain Rescue, and Disaster Rescue, with an option for "all" as a dataset reset. The user can also narrow the data set further by clicking the empty field at the top of the data set and inputting animal-specific information, like a name or age.

The Grazioso Salvare logo also needed to be present and link the user to the Salvare website. The choice to display the logo in the top right corner of the application seemed fitting for this type of project.

The final project also includes a graph showing the user the breeds of dogs and ages in weeks for the dataset loaded. This graph changes with the dataset, so if the user is specifically looking for a Water Rescue dog named Mika, this will return one result, breed Lab. Implementing a graph that displayed the breeds and age seemed like a good way to help the user streamline their query into the Salvare database.

Much like the first iteration of this application, there is a geolocation function that allows the user to track the first dog displayed on the dataset. If clicked, the pin will show the dog's name being tracked at any moment. This map will update as the user narrows their search, allowing a user to track a specific dog by breed, name, and rescue training type.

**Functionality Demo:**

[ProjectTwo rundown.wmv](https://vimeo.com/739487447)

[](https://vimeo.com/739487447)

**Tools used for the program:**

* MongoDB
* Python
* PyMongo
* HTML
* Dash framework

MongoDB was used for this project because MongoDB stores data in JSON-like documents. This creates databases that are very scalable and flexible. MongoDB is great for making databases easy to access as well, and this is great for a user-friendly program. Mongo and Python integrate with each other very easily, so easily in fact that they are a default combination for many projects involving databases. This is largely because of the flexibility and data types used within MongoDB and the flexibility that Python boasts.

The Dash framework was chosen because it is a Python framework that provides a very powerful and easy-to-use library for analytical web applications. The Dash framework simplifies the development of data-driven applications in a way that makes the development of programs like this one relatively easy to do regardless of web development experience.

**Links to Libraries/Tools:**

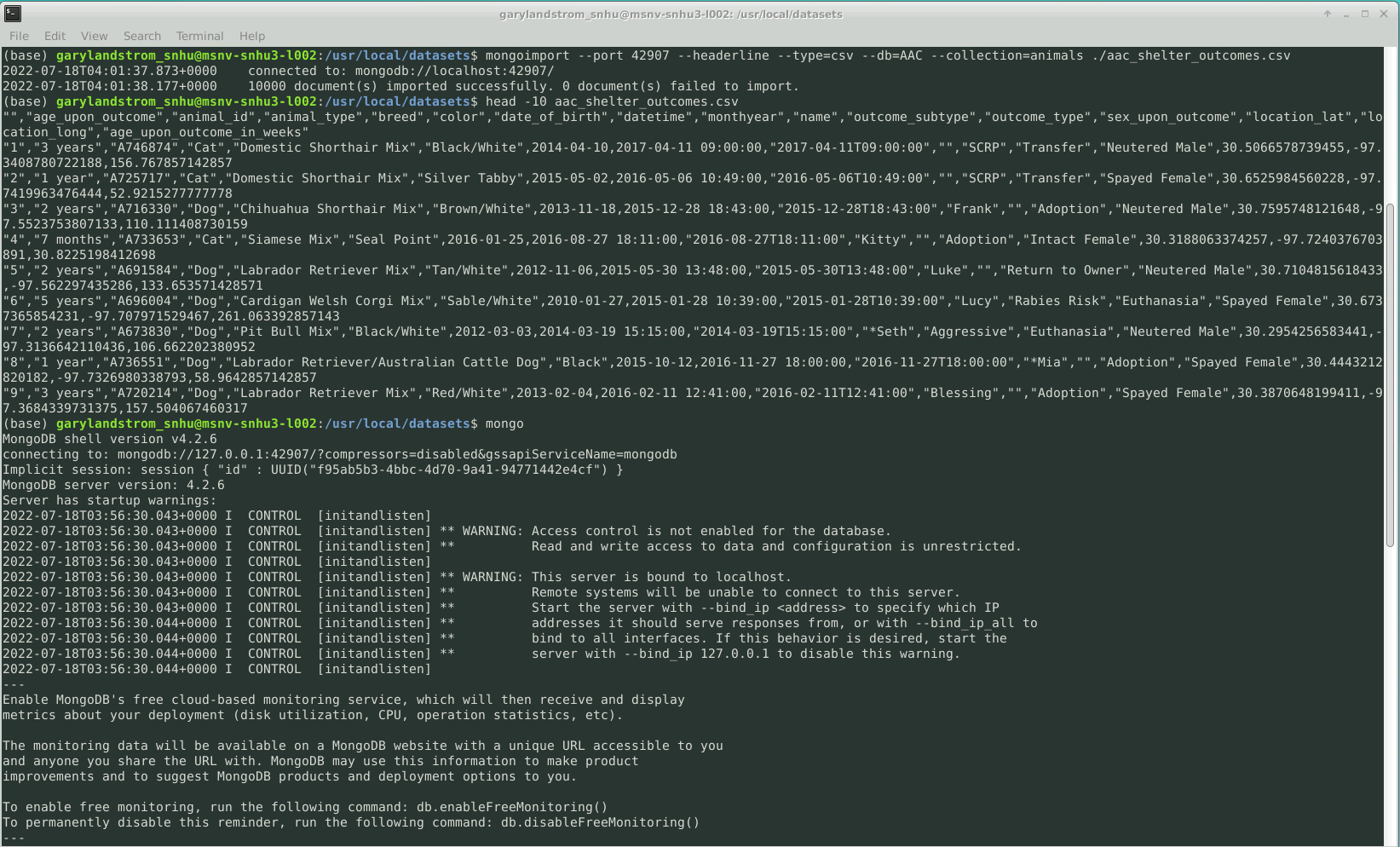
Each link will provide a comprehensive guide to installing and running properly. This project was created using a Linux-based terminal with MongoDB for the database. The Jupyter Notebook was used for the user interface and Python scripts.

* [**https://www.mongodb.com/**](https://www.mongodb.com/)
* [**https://www.python.org/**](https://www.python.org/)
* [**https://pypi.org/project/pymongo/**](https://pypi.org/project/pymongo/)
* [**https://plotly.com/dash/**](https://plotly.com/dash/)
* [**https://jupyter.org/**](https://jupyter.org/)
* [**https://www.linux.org/**](https://www.linux.org/)

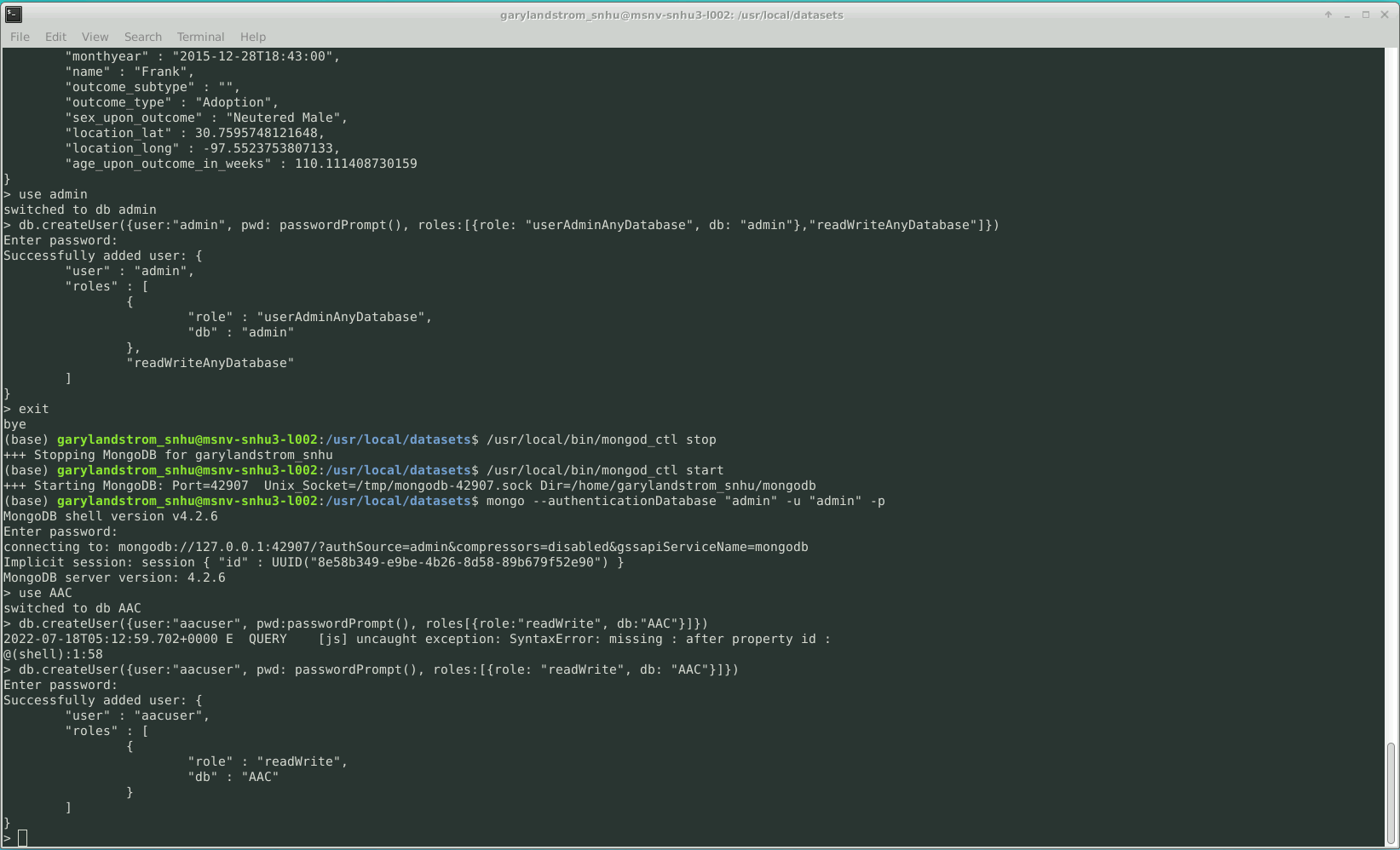
**Steps Taken:**

* MongoDB and Python were installed on a Linux based machine, this created the environment needed.
* The AAC database was imported to MongoDB and user accounts were established.

[import AAC]



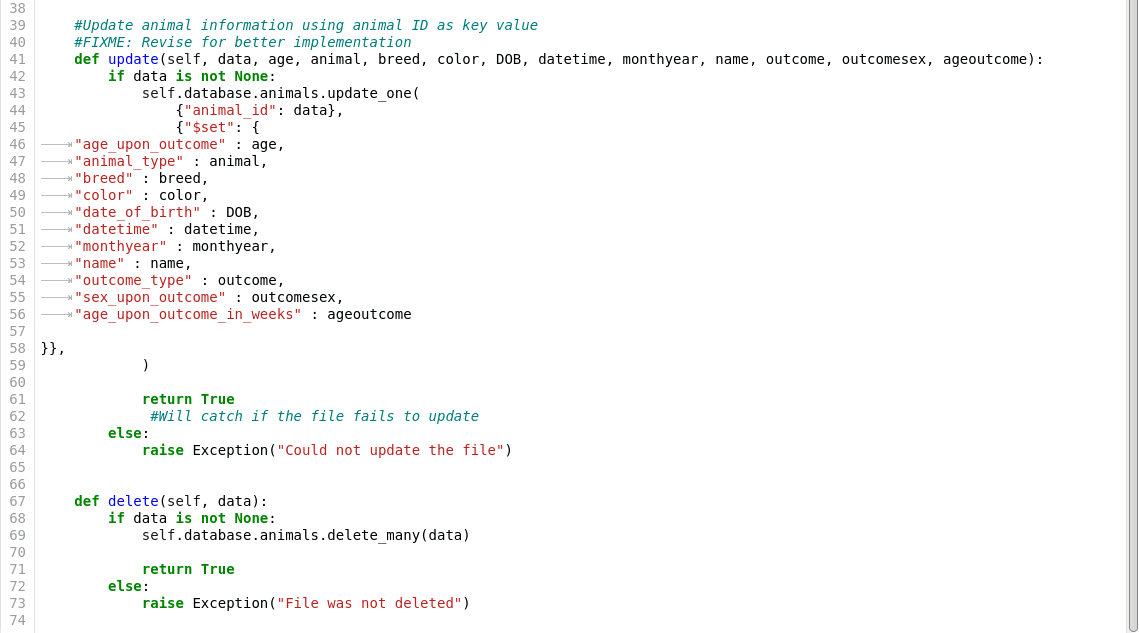
[User accounts]



* The main .py code was developed in Jupyter, this functionality included create, delete, update, read, and read\_all.

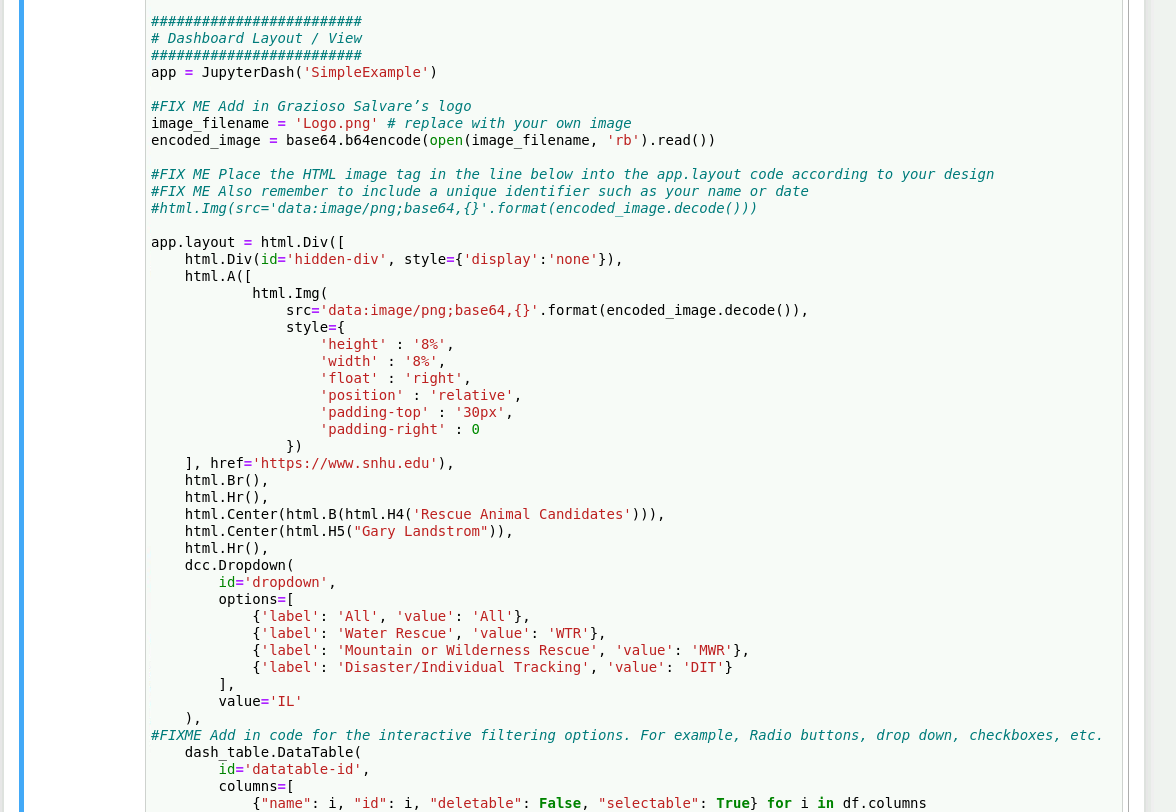
[.py main]





* The .ipynb file was created to provide the user with an easy-to-use format. This includes HTML and Dash functions to create a web-based application that can interact with the AAC database.

[.ipynb file]



**Challenges:**

Prior to this project, I worked with HTML, but it has been a long time since then. The most challenging aspect of this assignment was the HTML, primarily because it is easy to break and causes bugs within the program. Simply indenting incorrectly is enough to cause the program to fail or just fail to load part of the page. It was a learning curve getting back up to speed with HTML and its functions.

**Contact**

Gary Landstrom

[gary.landstrom@snhu.edu](mailto:Gary.landstrom@snhu.edu)

SNHU