**Grazioso README**

**OverView:**

For this readme file, I was working for Global Rain on creating a piece of software for Grazioso Salvare. The piece of software being created was for a software that can help Grazioso Salvare parse through existing data from animal shelters to identify and categorize available dogs. For this we have gone to contract full stack development for this application. Additionally this includes a database, and a client facing web application dashboard through which users at Grazioso Salvare can access the database. Additionally with the creation of this application, Grazioso will be able to place more resources into training and development of the dog.

**Background:**

For this project, the main thing to approach was to find an easy and streamlined solution that would benefit Grazioso. For the main premise we first need access to some animal shelters that are found locally to Grazioso. For the animal shelters we used the AAC database in order to find current up to date information about currently caged animals and use this as our basis for information. In order to incorporate the information in a precise manner, we used the MongoDB tool due to the effective storage and modification of data. Additionally, the use of MongoDB allows the utilization of python scripts containing standard CRUD functions for the database. Since there were certain requirements for the application, we had to find the best way to integrate it with the existing database infrastructure. Additionally within the python script, we used the Pandas dataframe structure along with Dash for formatting and integration.

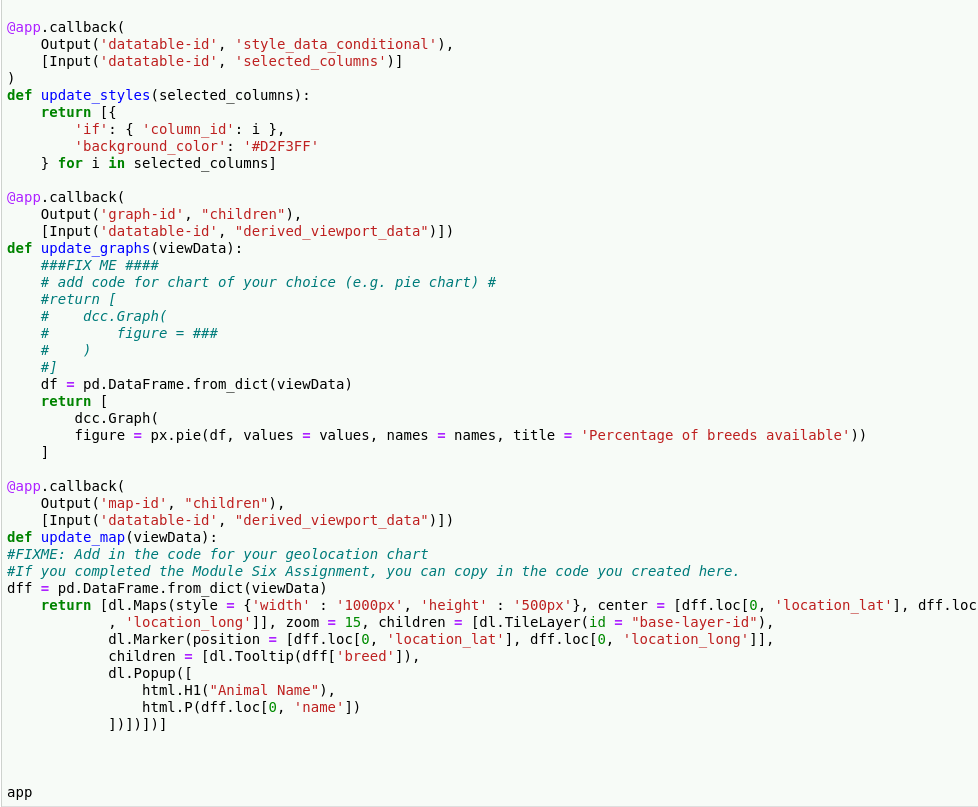
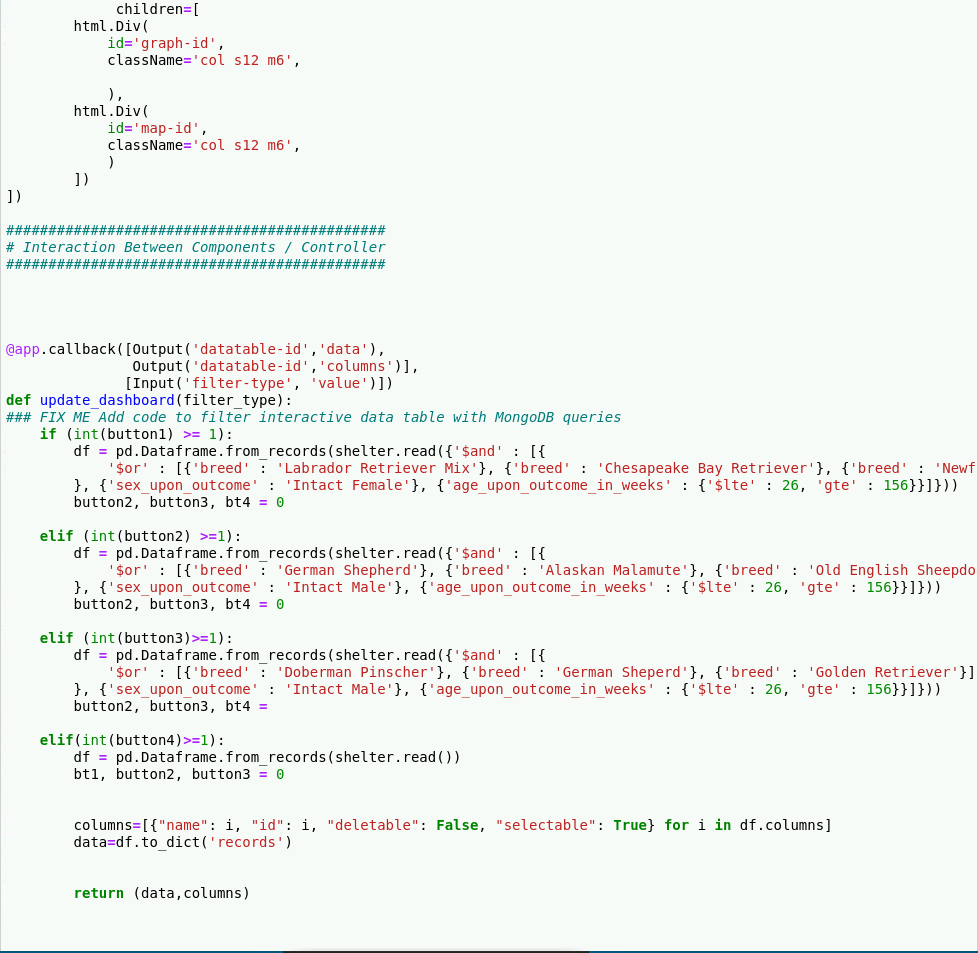
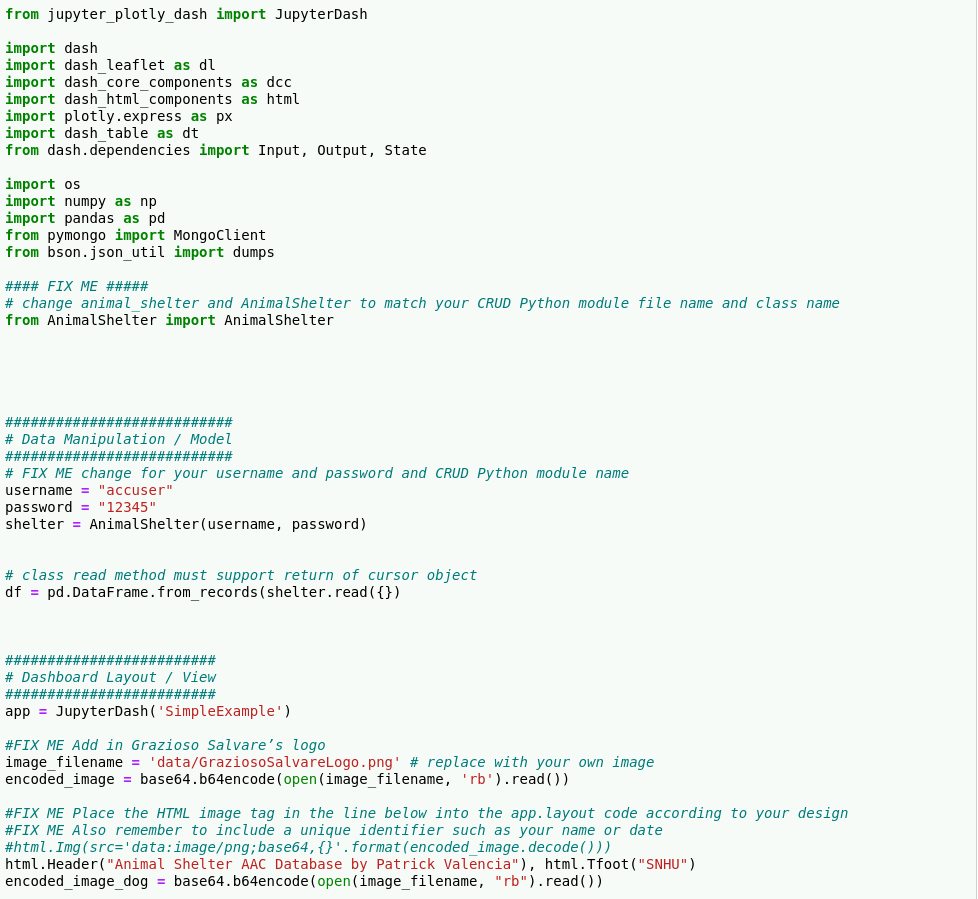
**Installation:**

Since this file is run through the Jupyter file manager, there is little installation required. The main thing that must be looked out for is the support of up to date HTML elements and a stable network connection. The stable network connection is for suitable access to the mongo database as well as access to a linux based terminal with the up to date versions of Python API, MongoDB API, and the Dash Framework API.

**Usage:**

For the usage of this project the goals were given to use by the client Grazioso Salvare. Since the main issue was their team needing a specialized tool for getting geographical location as well as making sure each query was able to bring in the correct information towards the needed usage. For instance, several of the buttons were created in order to help the user control the dashboard widgets. Due to this there was an in depth show of the data tables and charts for the queries. Additionally, there was a modification in the dash table to be much more interactive for the users when forming queries as well as completing charts. The screenshots below show the additional code added within the python script and functionality in order to properly achieve and answer all of Graciosos' needs. The added interactive options to filter data was added as well in order to filter out Water Rescue, Mountain or Wilderness Rescue, Disaster or Individual Tracking, or to reset the overall widgets within the data table and charts.

**ScreenShots:**



**Challenges:**

The main challenges presented within this assignment was the overall thought process of trying to figure out how to exactly put into code the users wants and needs. Thanks to proper documentation on the areas and what they needed to accomplish, it was slightly easier to streamline the users' needs. Overall the code should work and display the proper information along with the proper requested customization buttons. The main issue is the continued problem of having both the crud file implemented as well as the project trying to call onto the database. Besides those aspects, each section should properly work and expressively show the desired data for the AAc database for Grazioso to utilize.

**Contact Information:**

Patrick Valencia

[patrick.valencia@snhu.edu](mailto:patrick.valencia@snhu.edu)