**Project Two README**

**Functionality**

The following project allows the user to import a list from a database to filter and display information associated with it. This example shows entries from a database that holds animal shelter information. After importing, you can use different styles to choose what information you want to display. In this example I chose to display radio buttons so when a user clicks on a radio it’ll display a list of requested dogs, along with a pie chart to show what the population looks like and a map to display the location of these dogs.

Dogs that are more suitable for mountain rescue are on display.

A screenshot of a computer

AI-generated content may be incorrect.

Disaster rescue dogs are displayed when the radio button is pressed

A screenshot of a computer

AI-generated content may be incorrect.

If you don’t want to display any specific dogs you can simply push the reset radio button to display all animals, not just dogs.A screenshot of a computer

AI-generated content may be incorrect.

This can work with any species of animal that is currently in an animal shelter.A screenshot of a computer

AI-generated content may be incorrect.

**Tools**

**Mongo DB**

MongoDB is an easy-to-use and flexible database model that can either be self-hosted or hosted on a external location. It works in the backend and can easily connect to a front-end application via pymongo and Dash. Can be installed via the shell

**Dash**

Dash is a front-end application that offers widgets customized to your needs. Widgets like geolocation to show geographical information or pie charts to breed populations are just one of the many widgets available. Dash can be easily imported by importing Jupyter Dash if you’re using Jupyter Notebook

**Pymongo**

Pymongo is used to connect an application with the database, the glue that holds the communication between the two. It’s easy to implement by just importing it.

**Additional Tools**

**Jupyter Notebook**

If using Linux, you simply input the following command:



You can either click on the program to open it or type the following command:

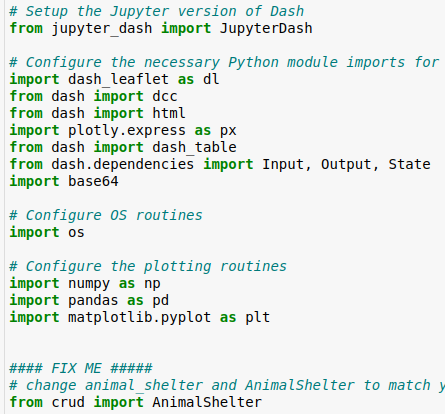


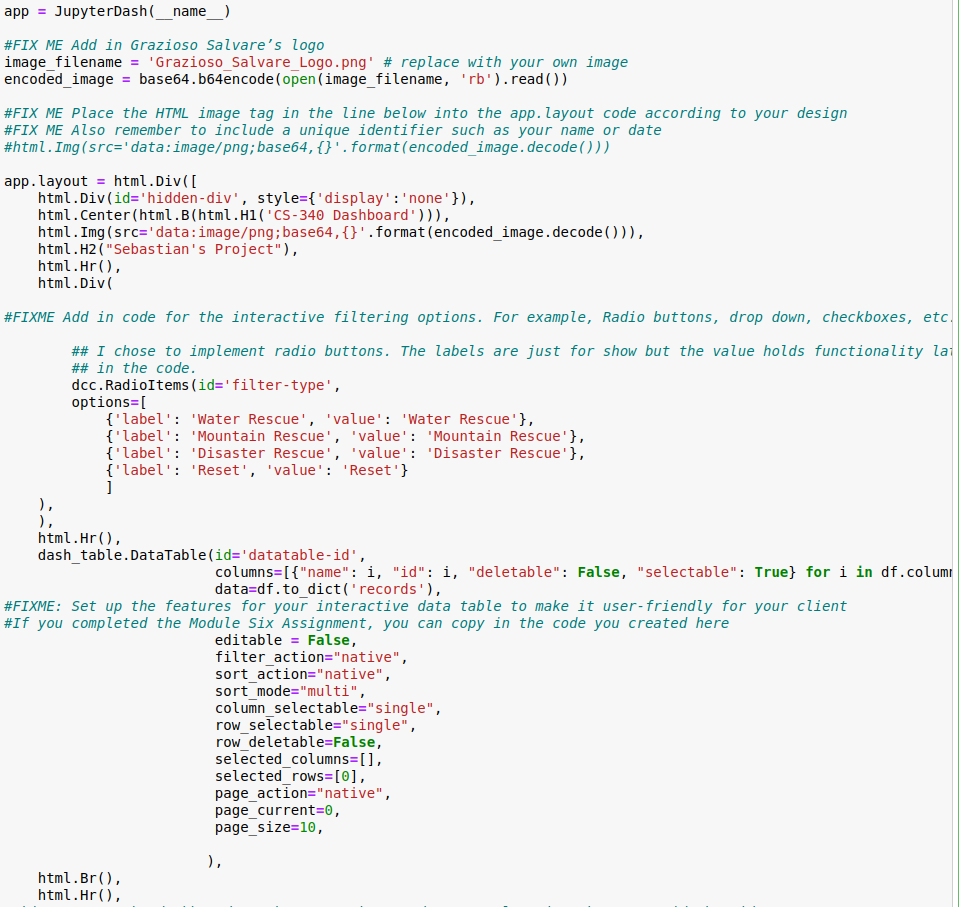
**Spyder**

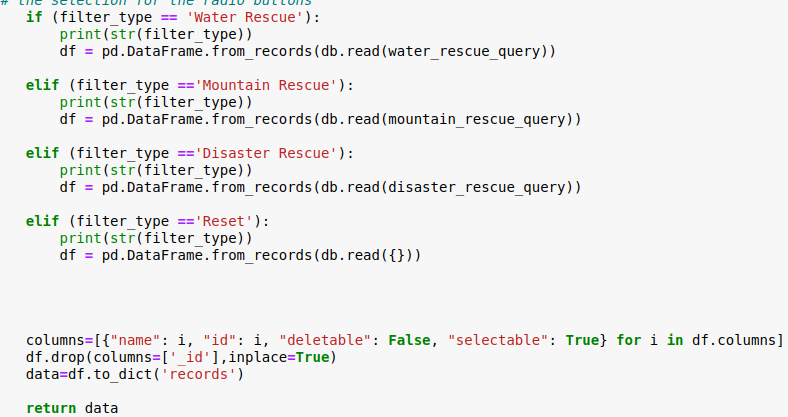
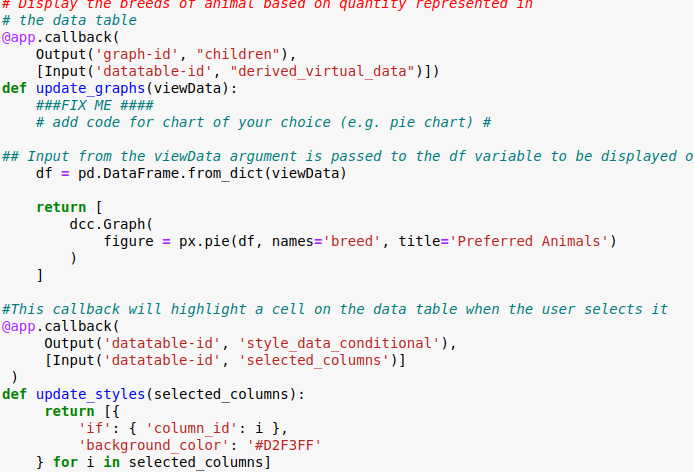
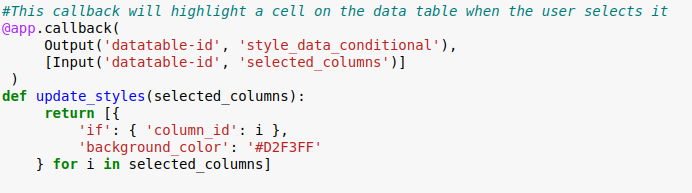
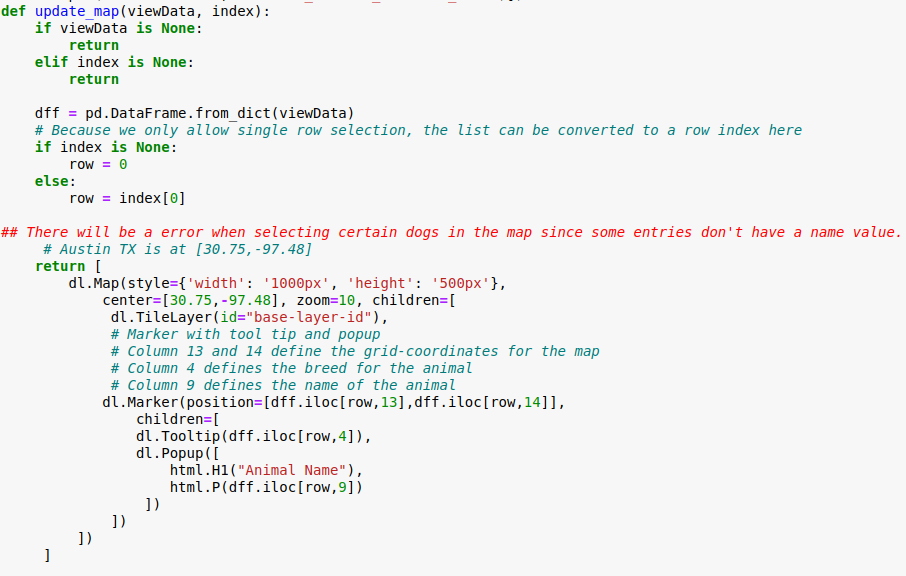
An IDE that’s simple to use

[Spyder Download Link](https://www.spyder-ide.org/download)

**Steps**

* To start, make sure you have all the tools available which above.
* Most work will be done in Jupyter notebook.
* Import the necessary dependencies along with the python module
  + ****
* Make sure you store your personal login information to the variables and pass them to the animal shelter constructor as you create an object.
  + **A close-up of a computer screen

    AI-generated content may be incorrect.**
* Call upon the read method and store the list in df
  + ****
* Next write the main body of the dashboard where all your widgets will be stored
  + ****
* Next write the function that’ll receive specific queries based on your taste so go ahead and replace the values with any information that you wish to retrieve and don’t forget about the callback. It’ll take the information that you give it and it’ll deliver it to the widgets.
  + **A screenshot of a computer code

    AI-generated content may be incorrect.**
  + ****
* Next up write the method used to update a pie graph by receiving information that you sent out from the previous method
  + ****
* The update style method highlights a cell that was clicked on
  + ****
* The final piece is the map. Again, it receives information fed to it by you the query functions and displays the location of the animals using longitude and latitude values.
  + ****

**Challenges**

* First, it’ll be confusing to see how all these components connect and fit together. Just remember MongoDB is the backend, Dash is the frontend, and pymongo is the glue that holds it together along with the python file module.
* Methods are already difficult enough and callbacks make it even more complicated. Callbacks are there to communicate with the widgets to give them the proper information to display so you can sit back and make them do all the work.
* One more thing, if the queries aren’t returning the information that you want you might want to check the spelling, programming in general requires precise syntax and grammar so don’t give up.