**Grazioso Salvare Readme**

# 

## About the Project/Project Title

Title

Grazioso Salvare Animal Shelter Dashboard.

Overview

This project uses a mongo database as the backend data source to provide information to a dashboard using the Dash framework in python. The dashboard includes a data table that displays the data located in the mongo database, a pie chart that dynamically changes based on the filters used in the data table, and a geolocation map that also dynamically changes based on the selected row in the data table.

MongoDB

MongoDB was used in conjunction with a python module that can manipulate and read items from the mongo database based on a query that is passed to one of the module’s CRUD (create, read, update, and delete) functions. MongoDB was chosen as the model component of development due to its ability to use these CRUD functions effectively in large databases such as this one.

Dash Framework

The Dash Framework was used to construct the dashboard itself and the different components located on the dashboard. This framework uses a HTML based formatting for its layout, and includes many different widgets that can be used to create visual representations of the data. The Dash components can also be used with plotly graphs to aid in these visual representations.

For this project, the **dash\_table** module was used for the data table. This data table was created from a pandas data frame,

The **plotly.express** module was used to create the pie chart.

The **dash\_leaflet** module was used for the geolocation graph.

And the **dash\_core\_components** module was used for widgets such as the dropdown menu in the app.

The documentation for these modules are linked below.

<https://dash.plotly.com/datatable>

https://plotly.com/python/pie-charts/

<https://leafletjs.com/reference.html#map-example>

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

## Installation

The tools that are needed can be installed are:

* MongoDB – can be found at <https://www.mongodb.com/docs/manual/installation/>
* The CRUD Python Module which can be downloaded here: ***C:\Users\metam\SNHU Classes\CS-340\Project 1\main.py***

(This is just a local path on my PC. I would typically use a link that would allow the user to download the file.)

* PyMongo library which can be found either on their website at <https://pypi.org/project/pymongo/> or

***pip install pymongo***

* Dash Framework that can be installed by following the instructions on their website at <https://dash.plotly.com/installation>

This module works best in jupyter-dash, so use

***pip install jupyter-dash***

* Dash-leaflet for the geolocation map <https://pypi.org/project/dash-leaflet/> or

***pip install dash-leaflet***

* Pandas to create the dataframes <https://www.pythoncentral.io/how-to-install-pandas-in-python/> or ***pip install pandas***

## How to Use the Dashboard

Each component of the dashboard can be used to interact with. A screen cast has been included here to demonstrate. Double click to open if viewing as a word document.



The Data Table

The data table has filtering, sorting, and selection options attached to it. The main filter that can be applied is to filter by rescue type. This can be done in the dropdown menu at the top of the table.

Graphical user interface, application, table

Description automatically generated

This filter will filter the data table to match the optimal rescue animals based on statistics collected by the Grazioso Salvare company.

The data in each column of the table can also be filtered based on a text field located at the top of each column. Note that capitalization does matter.

Additionally, each column can be sorted alpha-numerically either from first-to-last or last-to-first. This is done by clicking on the sorting arrows located next to each column’s name on the first row of the table.

The Pie Chart

This pie chart changes dynamically based on what rescue filter was applied to the data table. The chart shows the frequency of each breed type once the filter is applied. Below is a screenshot of each chart based on the rescue type selected from the dropdown menu.

Chart, sunburst chart

Description automatically generated A picture containing text, iPod

Description automatically generated

Chart, pie chart

Description automatically generated Chart, pie chart

Description automatically generated

Each of these charts can be filtered further by toggling the labels on/off which will either include or exclude that breed in the chart.

The Geolocation Map

The map will show the location of the animal that is selected on the datatable. To select an animal, simply click the radio button at the beginnning of the animal’s row. This will update the map to the selected animal’s information and place a marker on the map for the animal’s latitudal and longitudal coordinates.

Graphical user interface, application

Description automatically generated

## Completing the Project

There were several steps that were taken to complete this project.

1. Setting up Python CRUD module with mongoDB database

An outline of this process can be found in the README file located here.



1. Implementing the data from the database into the dashboard

This was done by first instantiating the CRUD python module into the dashboard application file and setting up an AnimalShelter object for the dashboard to use. Then, using different queries and the CRUD module’s read function, I was able to locate the different data sets needed for the widgets used in the application. Here is the code for this process.

Graphical user interface, text, application

Description automatically generated

1. Create the layout of the dash

This needed to be done first to layout a place for each component of the dash. Below is the code that is responsible for the layout. Notice that Dash uses html formatting to develop this layout.

Text

Description automatically generated with medium confidenceScatter chart

Description automatically generated

Graphical user interface, text, application

Description automatically generated

A picture containing graphical user interface

Description automatically generated

1. Create the select query function

To filter the data table by rescue type, queries needed to be made based on the Grazioso Salvare company’s statistical information. Because these queries were used by multiple components of the application, I made a function that could be referenced when the queries were needed by the different components (i.e. the data table and the bar chart).

1. Create the Data Table

This step simply used a callback function to determine what data was needed for the Input and outputs of the data table. In this case the input was the value selected in the dropdown rescue type filter and the output was the filtered data table. The data table was filtered using the select\_query function outlined in the previous step.

1. Create the Pie Chart

This was done using plotly.express’s pie method. The same query that was used to filter the data table in the previous step was used to select the data used in this pie chart.

1. Create the Geolocation Map

This map required a callback function with two inputs, the data from the data table as well as the row that was selected from the data table. These two inputs were passed to the function that was wrapped by this call back to generate the dash-leaflet map and dynamically change the coordinates/information of the marker that was used in the map.

## Challenges of the Project

There were challenges at pretty much every part of the project. Initially, getting familiar with mongoDB and setting up user authentication for the database proved to be difficult. Then getting the python CRUD module to work in conjunction with this database set up in mongoDB took some getting used to.

After this, getting acquainted with the Dash framework and learning how to implement the different widgets in the dcc\_core\_components module was challenging. This challenge was compounded by the fact that the documentation on Dash’s website is for a newer version of the framework and finding proper documentation on this older version of Dash made things confusing.

Learning the way the plotly data visualization tools worked in conjunction with the Dash framework also was confusing at first.

The main way all of these challenges were overcome is by spending hours studying/finding the correct documentation on the different components, functions, and technologies used in this application. There was rarely a way to solve any of these problems without first becoming familiar with how the many moving parts of this application worked. If there is one thing I have learned from this project, it is that proper documentation is key.