**CS – 340 Module 7-2 Project Two**

**Grazioso Salvare Dashboard**

**README**

**Michael Reynolds**

**SUMMARY**

This Dashboard is written in Python with Jupyter Notebook for visualizing data.

**DEPENDENCIES**

* Pymongo

Access Python MongoDB API for database queries, a very Python friendly database utilizing dictionary friendly query responses. It’s a very fast framework to interface with MongoDB as well.

* jupyter-dash

Allowed Dash to be used in Jupyter Notebooks

* dash-leaflet

for various dashboard components etc.

* plotly

for various dashboard components etc.

* numpy

Plotting routines

* pandas

Plotting routines

* matplotlib

Plotting routines

**INSTALLATION**

Use the package manager pip to install dependencies via the command below.

pip install -r requirements.txt

**CLIENT REQUIREMENTS**

**Dashboard Branding**

To meet Grazioso Salvare’s branding requirements, the company has requested the inclusion of the following components somewhere on the dashboard:

* The Grazioso Salvare logo. The company has requested that this logo include a URL anchor tag to the client’s home page: [www.snhu.edu](http://www.snhu.edu).
* A unique identifier (text or image) containing your name. Grazioso Salvare would like to credit you as the creator of the dashboard.

**Required Dashboard Widgets**

Grazioso Salvare is requiring the following widgets for the dashboard interface:

* Interactive filter options (buttons, drop-downs) to filter the Austin Animal Center Outcomes data set by:
  + - Water Rescue
    - Mountain or Wilderness Rescue
    - Disaster Rescue or Individual Tracking
    - Reset (returns all widgets to their original, unfiltered state)
* A data table which dynamically responds to the filtering options
* A geolocation chart and a second chart of your choice (such as a pie chart) that dynamically respond to the filtering options

**MEETING REQUIREMENTS**

The Grazioso Salvare logo is proudly displayed at the top center of the page, where it links to the client’s home page. Developer credits are displayed beneath it and the title for the dashboard. The dashboard exhibits effective data visualization, starting with the most prominent, a data table showing the filtered query results. Filtering is done via a drop down menu, where the options filter animal breeds by rescue type dynamically. Below this data table sports a pie chart showing breed distribution for the filter, as well as a geolocation chart showing the current selected data table row. By utilizing these elements/component, and ensuring proper use of a MVC pattern, we effectively queried/visualized/filtered data to the client’s needs, while also ensuring the page matches the mock-ups through UI/UX.

**Challenges Overcome**

By utilizing exhaustive tests, to determine any bugs/glitches, we were able to hash out any potentially harmful code, ensuring a stable deployment. Some of the issues that we had come across included query overloads, document to dictionary issues, and some element/component limitations. Through extensive debugging throughout the source code, a lot of these challenges were overcome. Many of these issues were related to non-viable approaches for the libraries/dependencies used in the source code.

**Testing Screenshots**

Logo

A logo of a dog

Description automatically generated with low confidence

Default Component States

A screenshot of a computer

Description automatically generated with medium confidence

Filter Component State

A screenshot of a computer

Description automatically generated with medium confidence