**README: Grazioso Salvare – Animal Rescue Dashboard**

**Author:** Godsgift Arokarawei  
 **Course:** SNHU CS-340  
 **Date:** June 17, 2025

**Project Overview**

This project involved the development of an interactive web dashboard for **Grazioso Salvare**, a company that specializes in animal rescue and training services. The primary objective was to create a dashboard that can display and filter shelter dog data to identify animals that are suitable for specific rescue training categories such as water rescue, mountain or wilderness rescue, and disaster or individual tracking.

Using a streamlined interface, users can filter animals by rescue type, view a data table of the results, and explore multiple visualizations, including breed distribution charts, outcome statistics, and geolocation maps showing the locations of rescued animals.

**Functionality Achieved**

All required features were successfully implemented in the dashboard. The application is capable of connecting to a MongoDB database to retrieve animal records in real time. It includes a dropdown widget that filters the data based on rescue type, dynamically updating a data table with matching results.

The dashboard also visualizes breed distributions using a pie chart, outcome types using a bar chart, and animal locations using an interactive map powered by Dash Leaflet. Additional branding elements, such as the Grazioso Salvare logo and student credit, were also incorporated. Testing was conducted, and the results were captured in screenshots to verify that each feature functions as intended.

Screenshots were taken to demonstrate the dashboard's key components. These include the full view of the dashboard on initial load, a filtered view showing animals qualified for water rescue, the breed distribution pie chart, the outcome type bar chart, and the interactive map with popup details on selected animals.

**Tools and Technologies Used**

This project was developed using **Python** and several specialized libraries. The **Dash** framework (by Plotly) was used to build the dashboard and provide a robust interface for connecting the data model with visual components. **Pandas** was used for data manipulation, while **Dash Leaflet** was integrated for geolocation visualization.

The database model was implemented using **MongoDB**, a NoSQL database that stores data in a flexible, JSON-like format. MongoDB was chosen because of its ability to efficiently store and retrieve semi-structured data, making it ideal for handling real-world records with variable schemas such as animal data. Additionally, MongoDB integrates smoothly with Python via the pymongo library, which allowed for fast querying based on multiple conditions (e.g., breed, age range, and sex).

The **Dash** framework served as both the view and controller in a traditional Model-View-Controller (MVC) pattern. Dash allowed for the construction of a clean and responsive user interface using HTML and CSS-like styling within Python. Callback functions connected user input to application behavior, creating a highly interactive experience.

**Implementation Steps**

To complete this project, I began by configuring a MongoDB connection using credentials provided in the course resources. I implemented a custom CRUD class to handle data retrieval. I then loaded the data into a Pandas DataFrame, performed cleaning and transformation tasks, and added simulated geolocation data where missing.

Next, I built the dashboard layout using Dash and added a dropdown filter for selecting the rescue training category. I created the filtering logic to apply specific criteria (breed, age range, and sex) for each rescue type, dynamically updating the data table based on the selected criteria.

Three key visualizations were added to the dashboard: a pie chart showing breed distribution, a bar chart showing the distribution of outcome types, and an interactive map displaying the geolocation of filtered animals. I also included branding with the Grazioso Salvare logo and personal attribution for the project.The app was tested thoroughly, and several screenshots were taken to document that each required feature functions correctly.

**Challenges and Solutions**

Several challenges were encountered during development. One of the first was related to the logo image not displaying correctly. This was resolved by verifying the image filename and ensuring it was placed in the assets directory, which Dash uses to serve static files.

Another issue was inconsistent or missing location data in the dataset. To solve this, I used NumPy to generate synthetic latitude and longitude values within a reasonable range. Finally, integrating multiple charts alongside a dynamic data table required careful layout management using Dash’s HTML components and responsive styles.

**Resources and References**

This project made use of several external resources. Documentation from Dash was critical in understanding layout and interactivity features. MongoDB's official site and Python tutorials for pymongo were also used to support database operations. For geolocation visualization, I relied on Dash Leaflet, which is a wrapper around Leaflet.js for Dash.

In addition, SNHU’s course-provided materials guided the implementation, particularly in the structuring of rescue filtering logic and the CRUD class setup.

**Final Notes**

This dashboard demonstrates a full-stack approach to building data-driven web applications using Python. By combining MongoDB’s flexible data storage, Dash’s powerful UI capabilities, and various Python libraries, the application fulfills all requirements and offers a usable interface for Grazioso Salvare to improve its rescue operations. All required features are functional, and the dashboard is ready for deployment or further extension.