# Grazioso Salvare Dashboard README

**Animal Records Dashboard**

This project involves the development of an interactive dashboard for Grazioso Salvare, a non-profit organization focused on animal rescue and adoption. The purpose of this dashboard is to allow users to filter, view, add, update, and delete animal records that are stored in MongoDB. The dashboard allows users to interact with the data through widgets, which filter the records based on the type of rescue, such as Water Rescue, Mountain or Wilderness Rescue, and Disaster or Individual Tracking. There is also a reset feature to clear all filters and return the dashboard to its original state.

**Required Functionality and Testing**

The dashboard functionality includes an initial view with widgets for filtering data, an interactive data table to display the records, and various charts to visualize the data. The filters include options for selecting different types of rescue operations, and once a filter is applied, the dashboard updates to display only the relevant records. The reset button clears all the filters, restoring the view to its original unfiltered state.

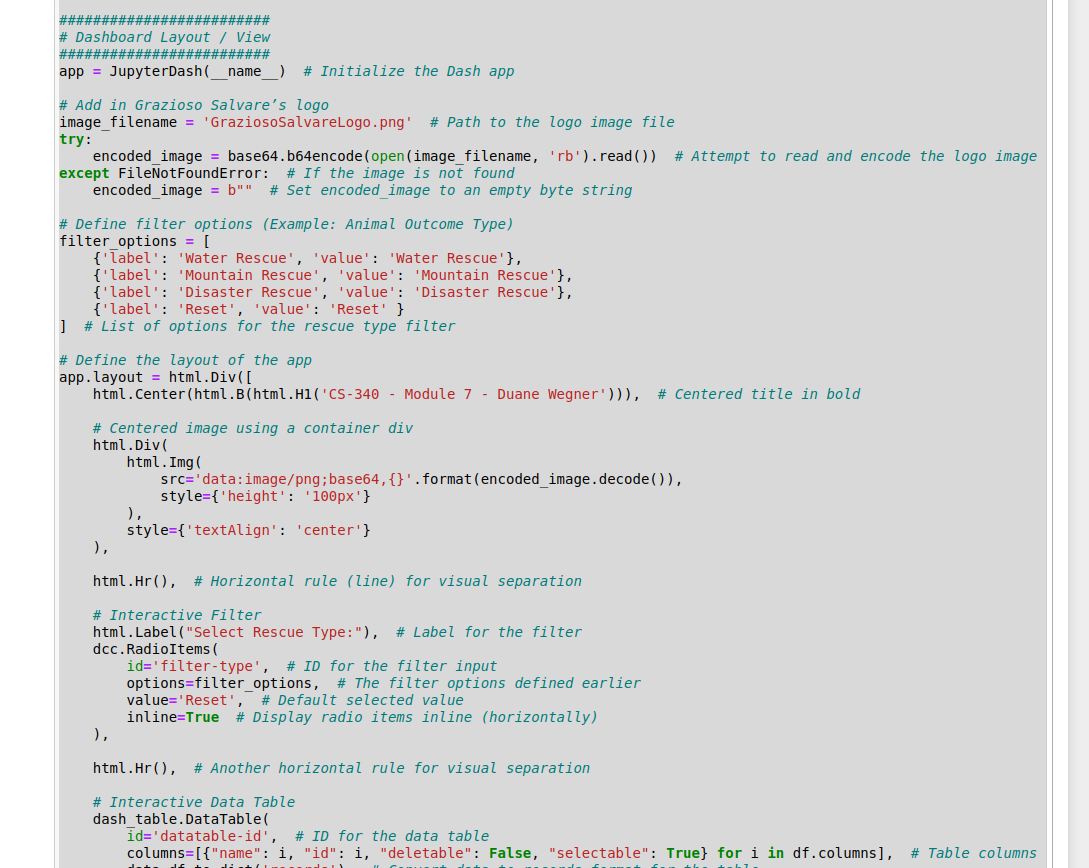
Screenshots have been included to demonstrate its functionality. The screenshots show the initial state of the dashboard, where all widgets are present, followed by examples of how the dashboard behaves after applying the different filters, such as Water Rescue, Mountain or Wilderness Rescue, and Disaster or Individual Tracking. The final screenshot or screencast demonstrates the reset action, which returns the dashboard to its original, unfiltered state.

**Tools and Technologies Used**

The dashboard is built using MongoDB as the database for storing animal records. MongoDB was chosen because it is a NoSQL database that offers flexibility in storing records with varying attributes, such as different types of animals and their unique needs. PyMongo, a Python library, is used to interact with MongoDB, making it easy to query and manipulate the records.

For the development of the interactive dashboard, Python and Jupyter Notebook were used, providing a convenient environment for data analysis and interactive development. Using Jupyter, a Python framework, was used to create the web interface and manage the frontend, allowing easy creation of interactive components such as dropdowns and buttons. Dash handles the view and controller aspects of the application, enabling real-time data updates in response to user interactions.

The dashboard is built to display animal records from MongoDB and allows users to filter the records based on the type of rescue. A simple dropdown menu enables the user to select a rescue type, and the table updates to reflect only the records matching that filter. Below is an example of the code used to achieve this:



A screenshot of a computer program

AI-generated content may be incorrect.

This snippet demonstrates how the dashboard dynamically updates based on the user's selection from the dropdown menu.

**Challenges and Solutions**

During the development of the dashboard, several challenges were encountered. One challenge was performance-related: the queries to MongoDB were slow when dealing with large datasets. This was addressed by indexing key fields in the database to speed up queries.

**Steps to Complete the Project**

1. **Set Up MongoDB**: MongoDB was installed and configured on the local machine. A new database called animal\_db was created, and sample data was inserted into the animals collection.
2. **Develop Backend Logic**: Python was used for backend logic to handle queries to MongoDB, allowing the dashboard to retrieve and display animal records based on the user's selected filters.
3. **Create Frontend with Jupyter**: Jupyter Dash was used to create an interactive frontend that allows the user to filter animal records by rescue type. Dropdown menus and buttons were added to trigger updates to the data displayed on the dashboard.
4. **Test the Application**: The dashboard was thoroughly tested to ensure that the data filters work as expected and that the reset button clears all applied filters.
5. **Document the Project**: This README file was created to provide documentation for the project, explaining the functionality, tools used, and the development process.

**Resources and References**

* [MongoDB](https://docs.mongodb.com/)
* [Jupyter Notebook](https://jupyter.org/)

**Conclusion**

The Animal Records Dashboard successfully meets the needs of Grazioso Salvare by providing an interactive and user-friendly interface to manage animal rescue data. The dashboard allows users to filter records by rescue type, view the results in an interactive table, and reset the filters to view all records. This project demonstrates the power of using MongoDB with Python and Jupyter to create an intuitive, data-driven web application.

### Screenshots

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a map

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

^ Shows age being filtered from youngest to oldest in the reset state ^

A screenshot of a computer

AI-generated content may be incorrect.

## Contact

Duane Wegner: [duane.wegner@snhu.edu](mailto:duane.wegner@snhu.edu)

**Updated**

April 14, 2025