# **README: Web Application Dashboard for Grazioso Salvare**

## **Project Overview:**

This project implements a **Web Application Dashboard** for **Grazioso Salvare**, a rescue-animal training organization. The dashboard allows users to interact with and visualize data from the Austin Animal Center, filtering and displaying animal shelter outcomes relevant to the organization’s needs. It is developed using the **Dash** framework, **MongoDB** as the database, and **Python** for the backend. This dashboard helps identify dogs suitable for search-and-rescue training.

## **Purpose of the Dashboard**

The dashboard aims to:

* Provide an intuitive interface for Grazioso Salvare to access and filter animal shelter data.
* Display data dynamically in response to user queries, helping to identify dogs suited for various rescue types (water rescue, mountain rescue, etc.).
* Visualize data with interactive tables and charts to support decision-making.
* Provide an open-source solution that can be used by similar organizations.

## **How to Use the Dashboard**

### **1. Setting Up the Environment**

Before running the dashboard, ensure you have the following prerequisites installed:

* **Python 3.x** installed.
* **MongoDB** installed and running, with the appropriate data loaded.
* Required Python packages installed via pip:
  + pip install dash pandas pymongo plotly

### **2. Importing the Dashboard Code**

To run the dashboard, clone the project repository or use the provided project files (ProjectTwoDashboard.ipynb).

### **3. Connecting to MongoDB**

The MongoDB instance should be up and running. The connection is made through the pymongo driver in the dashboard code. Here is how the connection is set up in the code:

from crud import AnimalShelter # Assuming you imported the Project One CRUD module  
username = "username"  
password = "password"  
host = "mongodb"  
port = 35180  
db\_name = "AAC"  
collection\_name = "animals"  
shelter = AnimalShelter(username, password, host, port, db\_name, collection\_name)

### **4. Running the Dashboard**

Once the environment is set up, run the Jupyter Notebook (ProjectTwoDashboard.ipynb) to launch the interactive dashboard. Use the following command in a Jupyter cell to run the dashboard:

!jupyter notebook ProjectTwoDashboard.ipynb

This will open the dashboard in your web browser.

## **Dashboard Features and Functionality**

The dashboard offers various interactive components that enable users to filter animal shelter data based on different criteria. The key features include:

### **1. Interactive Data Table**

The data table dynamically displays the available animal shelter records. Users can filter the data based on different rescue types or preferred dog breeds. The table also supports features such as pagination and sorting.

### **2. Filtering Options**

Users can filter the displayed data by selecting the following options:

* **Rescue Type** (Water Rescue, Mountain or Wilderness Rescue, Disaster or Individual Tracking)
* **Preferred Dog Breed** (Labrador Retriever, German Shepherd, etc.)

### **3. Data Visualizations**

The dashboard includes the following charts:

* **Geolocation Chart**: Displays the location of animal shelters.
* **Rescue Type Distribution Chart**: A bar chart showing the distribution of different rescue types among the available dogs.

### **4. Dashboard Interactivity**

The filters on the dashboard are interactive, updating the data table and visualizations in real-time. When a user selects a rescue type or breed from the dropdown menus or radio buttons, the displayed data and charts are automatically updated to reflect the selection.

## **Tools and Technologies Used**

### **1. MongoDB**

MongoDB was used as the database for this project due to its flexibility in handling large datasets and its capability to store unstructured data. MongoDB’s NoSQL structure allows for easy querying and efficient retrieval of shelter data, which is crucial for this project.

### **2. Dash Framework**

The **Dash** framework was used for the front-end of the dashboard. Dash allows for the creation of highly interactive web applications with Python. It integrates easily with **Plotly** for creating dynamic charts and **pandas** for data manipulation. Dash’s simplicity and flexibility make it an ideal choice for creating interactive web dashboards.

### **3. Pymongo**

**pymongo** was used to interact with the MongoDB database. It provides an easy-to-use interface for querying, inserting, updating, and deleting records in the database.

### **4. Plotly**

**Plotly** was used to create interactive charts for visualizing the data, including the geolocation and rescue type distribution charts.

## **Steps to Complete the Project**

The following steps were followed to build and deploy the dashboard:

1. **Data Retrieval and Setup**: Used the CRUD module from Project One to retrieve data from MongoDB.
2. **Dashboard Design**: Designed the layout using Dash’s HTML and Core Components.
3. **Interactivity Implementation**: Added interactivity to the dashboard through callbacks that update the data table and charts based on user input.
4. **Testing and Debugging**: Ensured the dashboard works by testing it with different filters and making sure all components respond correctly.
5. **Documentation**: Wrote this README to document the setup, usage, and functionality of the dashboard.

## **Challenges and Solutions**

* **MongoDB Connection Issues**: At the beginning, there were some issues with connecting to MongoDB on the specified host and port. After ensuring the correct credentials and MongoDB configuration, the connection was successfully established.
* **Interactive Components**: Initially, the interactivity between the data table and the charts was not functioning as expected. By carefully setting up Dash callbacks to bind the input from the dropdown and radio components to the relevant chart and table updates, the interactivity was implemented successfully.
* **Data Filtering**: Filtering data by breed and rescue type required multiple queries to MongoDB. These were optimized to ensure that the dashboard updated the data efficiently without performance issues.

## **Conclusion**

This project provides a fully functional, user-friendly dashboard for **Grazioso Salvare** to filter and visualize shelter animal data based on their needs for search-and-rescue training. The use of **MongoDB**, **Dash**, and **Plotly** provides a powerful, scalable solution for interacting with large datasets and creating interactive visualizations.