# Steven Croft’s Grazioso Salvare – Project Two Dashboard

## 1. Project Functionality

This project delivers a fully functional dashboard web application for Grazioso Salvare, enabling users to interact with and visualize the Austin Animal Center Outcomes dataset. The dashboard supports filtering by rescue type (Water Rescue, Mountain/Wilderness Rescue, Disaster/Individual Tracking), displays an interactive data table, and dynamically updates visualizations such as a geolocation map and a bar chart. A reset option returns the dashboard to its unfiltered state.

## 2. Tools Used

- Python: Primary programming language for implementation  
- Dash: Provides the web application framework (MVC design pattern)  
- Dash Leaflet: For geolocation mapping  
- Plotly: For interactive charting  
- Pandas: For dataset manipulation and processing  
- MongoDB: Used for CRUD operations (initially, but dataset fallback is also included)

## 3. Why MongoDB?

MongoDB was selected as the model component because it is a flexible NoSQL database that handles semi-structured data well, such as the AAC outcomes dataset. Its integration with Python via PyMongo allows for straightforward CRUD operations, enabling developers to query and filter data efficiently.

## 4. Dash Framework

The Dash framework provides both the view (dashboard widgets, charts, tables) and controller (interactive callbacks, event handling) of the application. It allows for a user-friendly, intuitive interface that dynamically responds to user input and reduces training time.

## 5. Steps Taken

1. Set up MongoDB database and CRUD Python module.  
2. Created an interactive data table for unfiltered AAC outcomes.  
3. Developed queries and filtering functions for rescue types.  
4. Built interactive widgets (dropdowns/radio buttons) to control filters.  
5. Created a geolocation map and bar chart that update with filters.  
6. Integrated the Grazioso Salvare logo and unique identifier.  
7. Tested functionality and captured screenshots.

## 6. Challenges and Solutions

- \*\*Connection Issues\*\*: MongoDB server access was sometimes refused. To resolve this, a local dataset fallback was added so the dashboard can still operate without live DB access.  
- \*\*Data Cleaning\*\*: Inconsistent data fields required preprocessing using Pandas.  
- \*\*Interactive Linking\*\*: Making sure all widgets responded to filters required careful setup of Dash callbacks.

## 7. Resources

- Dash Documentation: https://dash.plotly.com/  
- Plotly: https://plotly.com/python/  
- MongoDB PyMongo: https://pymongo.readthedocs.io/  
- Pandas: https://pandas.pydata.org/