Grazioso Salvare Dashboard — CS 340 Project Two (README)

Author: Chris Phillips  
Date: August 12, 2025

1) Project Overview

This project delivers a client-facing dashboard for Grazioso Salvare to identify dogs that meet specific search-and-rescue profiles. The app connects to a MongoDB database and provides:

* An interactive data table of shelter outcomes
* A geolocation map (Leaflet) that centers on the selected row
* A Top Breeds bar chart that reflects the current table contents
* Rescue-type filters (Water, Mountain/Wilderness, Disaster/Individual Tracking, Reset) that update all widgets

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Water Rescue

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Mountain/Wilderness Rescue

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Disaster/Individual Tracking

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Tools and Rationale

* MongoDB + PyMongo — Flexible document database that maps directly to Python dict/JSON, ideal for semi-structured shelter records and incremental schema changes.
* Python + Dash (JupyterDash) — Rapid development of interactive UIs in notebooks. Dash callbacks implement the controller logic; Dash components render the view.
* dash-leaflet — Integrates Leaflet maps in Dash for intuitive geolocation markers.
* Plotly Express — Quick, interactive charts with minimal code (used for Top Breeds bar chart).
* Pandas — CSV ingest/cleanup and convenient tabular operations.

Architecture (MVC Pattern)

* Model — MongoDB database aac, collection outcomes, accessed through a CRUD class AnimalShelter (animal\_shelter.py) with create/read/update/delete methods.
* View — Dash components: DataTable, dcc.Graph (bar chart), dash\_leaflet map, branding header, and radio filters.
* Controller — Dash callbacks that call the CRUD read() method to fetch filtered data and update all widgets.

Pattern sets:

* Water Rescue
  + animal\_type = Dog (case-insensitive)
  + sex\_upon\_outcome = Intact Female (case-insensitive)
  + age\_upon\_outcome\_in\_weeks ∈ [26, 156]
  + breed matches any of: Labrador Retriever, Chesapeake Bay Retriever, Newfoundland
* Mountain/Wilderness
  + animal\_type = Dog
  + sex\_upon\_outcome = Intact Male
  + breed matches any of: German Shepherd, Alaskan Malamute, Old English Sheepdog, Siberian Husky, Rottweiler
* Disaster / Individual Tracking
  + animal\_type = Dog
  + sex\_upon\_outcome = Intact Male
  + age\_upon\_outcome\_in\_weeks ∈ [20, 300]
  + breed matches any of: Doberman Pinscher, German Shepherd, Golden Retriever, Bloodhound, Rottweiler