README – Grazioso Salvare Dashboard Application

Author: Umar Asif

# Project Overview

The purpose of this project was to create a fully functional dashboard application for Grazioso Salvare, a company that trains rescue dogs. The dashboard connects to a MongoDB database and allows users to filter, view, and interact with data related to dogs available for training.  
  
The application uses a Model-View-Controller (MVC) pattern:  
- Model: MongoDB database for animal shelter data  
- View: Dash dashboard interface (charts, maps, tables)  
- Controller: Python CRUD module (AnimalShelter.py)

# Required Functionality

The dashboard meets the following requirements:  
- An unfiltered interactive data table showing all animals  
- Radio buttons to filter by:  
 • Water Rescue  
 • Mountain or Wilderness Rescue  
 • Disaster or Individual Tracking  
 • Reset  
- A geolocation chart that shows the location of a selected dog  
- A pie chart showing the breed distribution for filtered results  
- Grazioso Salvare logo (clickable link to SNHU) and author identifier  
- Fully dynamic interaction between filters, charts, and data table  
  
Screenshots were captured to demonstrate each of the above states and are included separately.

# Tools & Technologies Used

- Python 3  
- Dash – for building the interactive dashboard  
- dash\_leaflet – for the geolocation map  
- pymongo – to connect Python to MongoDB  
- MongoDB to store and query animal data  
- Jupyter Notebook – for developing and testing the dashboard

# Why MongoDB?

MongoDB was used because:  
- It stores data in a flexible document-based format (BSON/JSON).  
- It integrates easily with Python via pymongo.  
- It's ideal for semi-structured data like animal shelter records.  
- Queries can be dynamically passed as dictionaries.

# Why Dash?

Dash was used because:  
- It combines Flask (backend) and React.js (frontend) under one Python framework.  
- It simplifies building dynamic dashboards and charts.  
- It supports live callbacks for interactivity between filters and visuals.

# Steps Taken to Complete the Project

1. Reused and connected the AnimalShelter class from Project One.  
2. Loaded shelter data into a Pandas DataFrame.  
3. Built the base layout using Dash with:  
 - DataTable  
 - Leaflet map  
 - Pie chart  
4. Created interactive radio buttons for filtering by rescue type.  
5. Used Dash callbacks to connect filters to the charts and table.  
6. Tested the application with each filter and captured screenshots.  
7. Packaged the project with code, screenshots, and documentation.

# Challenges & How They Were Solved

- Logo not displaying: Fixed by placing the image in the correct assets/ folder.  
- Dash crashing on MongoDB ObjectId: Solved by dropping the \_id column.  
- Callbacks not updating charts: Added checks for None and ensured proper use of Input vs State.  
- Authentication to MongoDB failing: Fixed port and user/pass issues with Apporto config.