README

By Aqbah Butt

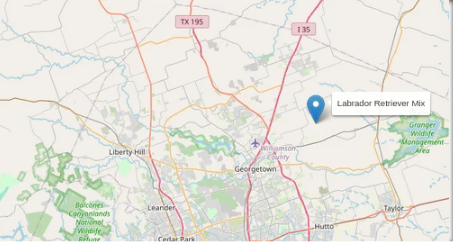
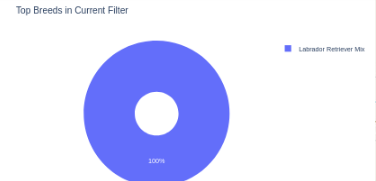
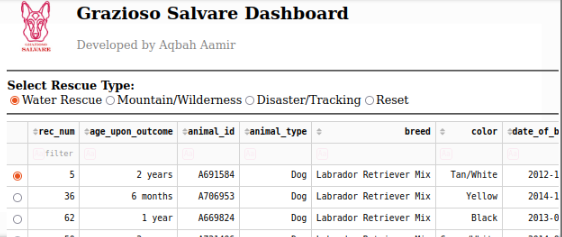
Overview of the Project

For Grazioso Salvare, a global rescue-animal training organization, this project provides a fully functional dashboard. The dashboard allows users to find dogs who would make suitable search-and-rescue training candidates by integrating with a MongoDB database that contains shelter data from the Austin Animal Center.

The Dashboard provides:

* Interactive filtering options to swiftly identify dogs based on rescue profiles (water rescue, mountain/wilderness, disaster/individual tracking).
* An interactive table of data that reacts to filters in real time.
* A geolocation map that uses the latitude and longitude of specific animals to plot them on a leaflet map.
* Breed distribution is dynamically updated by filter settings and displayed in a pie chart.
* The Grazioso Salvare logo and a unique identifier for the developer.

Screenshots of the Dashboard with different filters applied.

* Water Rescue  
  
* Mountain or Wilderness Rescue  
  A screenshot of a computer

  AI-generated content may be incorrect.A colorful circle with numbers and a circle in the middle

  AI-generated content may be incorrect.A map with a blue pin

  AI-generated content may be incorrect.
* Disaster or Individual Tracking  
  A screenshot of a computer

  AI-generated content may be incorrect.A pie chart with numbers and a circle

  AI-generated content may be incorrect.A map with a location pin

  AI-generated content may be incorrect.
* Reset  
  A screenshot of a computer

  AI-generated content may be incorrect.A colorful pie chart with numbers and a few different colored circles

  AI-generated content may be incorrect.A map with a location pin

  AI-generated content may be incorrect.

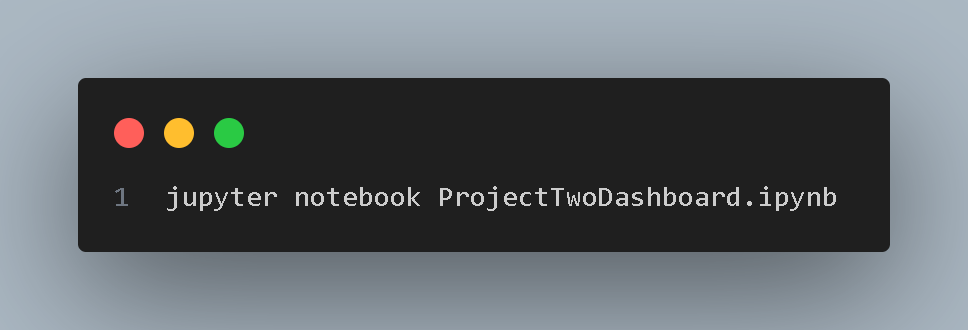
Tools Used

* MongoDB (Database – Model)
  + Chosen for its flexible, document-based storage format, which matches the JSON-like structure of animal shelter data.
  + Supports complex queries, fast indexing, and works seamlessly with Python via PyMongo.
  + Stores geolocation data (lat/long) used for mapping on the dashboard.
* Python & Dash Framework (View + Controller)
  + Dash provides a reactive framework for building data dashboards with minimal code.
  + Supports integration of tables (dash\_table), charts (plotly.express), and maps (dash\_leaflet).
  + Callback structure ensures MVC separation: filters act as the controller, MongoDB queries provide the model, and widgets act as the view.
* Dash Leaflet
  + Used for rendering the geolocation map, providing an intuitive visualization of where the animals are located.
* Plotly Express
  + Used to create dynamic charts (pie chart of breeds).
* JupyterDash
  + Allows running the dashboard interactively in a Jupyter Notebook environment.

Project Setup and Steps Taken

1. CRUD Python Module (Project One)
   1. Developed a Python class (AnimalShelter) with full Create, Read, Update, delete functionality using PyMongo.
   2. Configured authentication with the aacuser account.
2. Dashboard Foundation (Module Five & Six Milestones)
   1. Implemented login/authentication to MongoDB from the dashboard.
   2. Built the interactive data table showing animal records.
   3. Added the geolocation map with single-row selection support.
3. Project Two Enhancements
   1. Implemented interactive filter options (radio buttons).
   2. Linked filters to MongoDB queries via callbacks.
   3. Connected the interactive data table to both the pie chart and geolocation map.
   4. Added the Grazioso Salvare logo and a unique identifier.
   5. Completed testing for each filter (Water Rescue, Mountain/Wilderness Rescue, Disaster/Tracking, Reset).

How to Reproduce

1. Clone the repository containing:
   1. ProjectTwoDashboard.ipynb
   2. AnimalShelter.py (CRUD module)
   3. Grazioso Salvare Logo.png
2. Ensure the following Python packages are installed:  
   
3. Start MongoDB and ensure the AAC.animals collection is loaded with shelter data.
4. Update your MongoDB host, port, username, and password inside AnimalShelter.py.
5. Open and run the Jupyter Notebook:  
   
6. Interact with the dashboard:
   1. Select rescue type filters.
   2. View the updated table, pie chart, and geolocation map.
   3. Reset to show all data.

Challenges and Solutions

* Challenge: Initial crashes in the DataTable caused by MongoDB \_id fields.
  + Solution: Dropped the \_id field in the DataFrame before rendering.
* Challenge: Map not rendering when no rows were selected.
  + Solution: Defaulted to showing the first row when no row is selected and added error handling for missing data.
* Challenge: Making filters intuitive for the client.
  + Solution: Implemented radio buttons with clear rescue-type labels, directly mapped to MongoDB queries.