Jeffrey Flores

February 22, 2025

CS-340 Project Two ReadMe

**CS-340 Animal Rescue Dashboard**

**Project Overview**

The CS-340 Animal Rescue Dashboard is a web-based dashboard designed for Grazioso Salvare to assist in identifying and categorizing rescue dogs from the Austin Animal Center dataset. The dashboard allows users to filter, view, and visualize dog data to aid in search-and-rescue training selection.

**Required Functionality**

This project implements the following required functionalities:

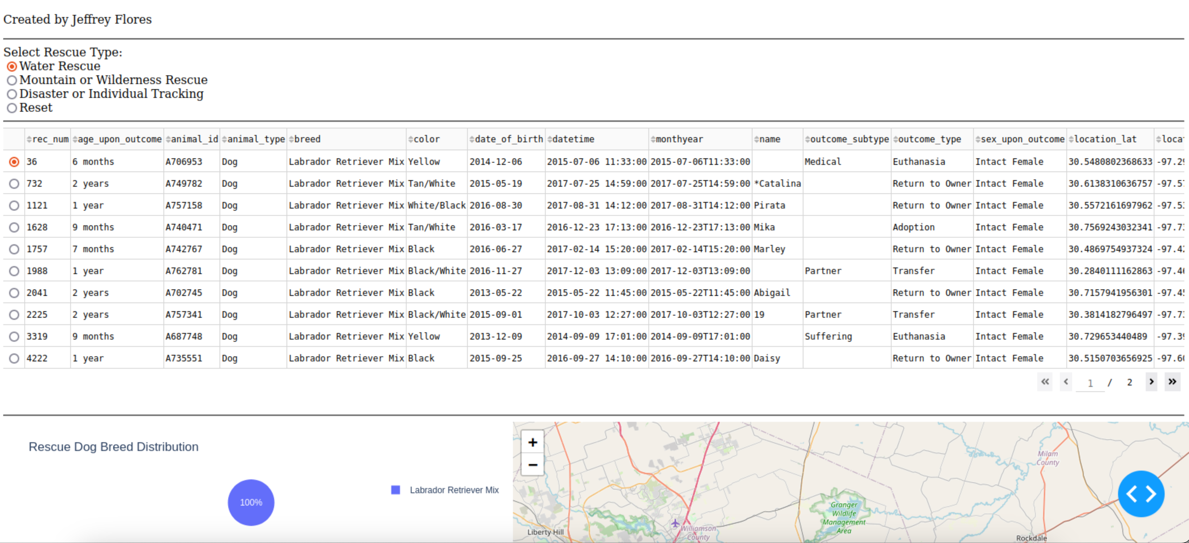
1. **Interactive Filtering Options:** Users can filter data based on different rescue types:
   * Water Rescue
   * Mountain or Wilderness Rescue
   * Disaster or Individual Tracking
   * Reset (shows all data)
2. **Dynamic Data Table:**
   * Displays real-time data from MongoDB.
   * Updates based on selected filter criteria.
   * Supports sorting, pagination, and row selection.
3. **Data Visualization:**
   * **Pie Chart:** Displays breed distribution for the selected rescue category.
   * **Geolocation Map:** Shows the exact location of selected dogs in the dataset.
4. **Branding:**
   * Displays the **Grazioso Salvare logo** (linked to [https://www.snhu.edu](https://www.snhu.edu/)).
   * Includes a **developer credit** (My Name).

**Screenshots of the Dashboard Functionality**

**1. Default Dashboard View**

A screenshot of a computer

AI-generated content may be incorrect.

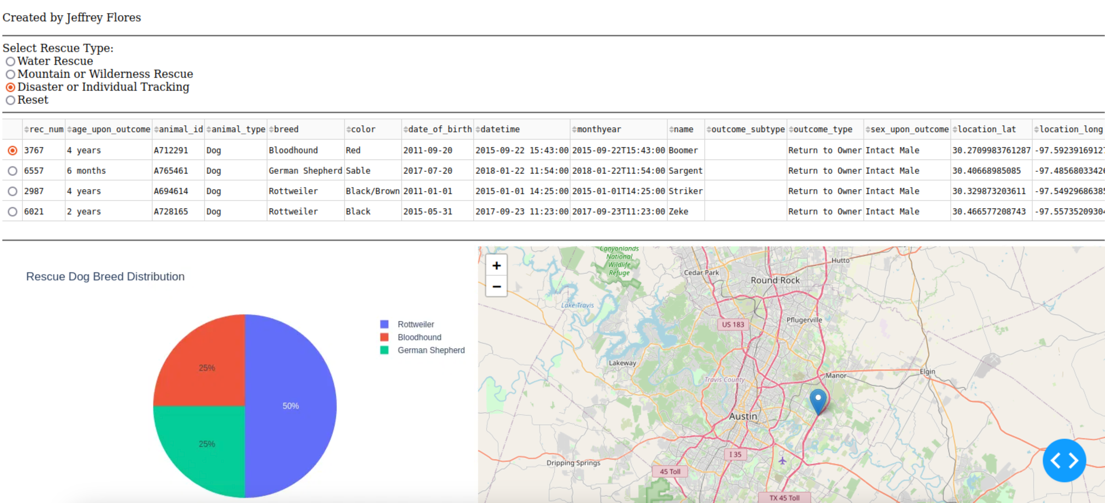
**2. Water Rescue Filter Applied**

**3. Mountain Rescue Filter Applied**

A screenshot of a computer

AI-generated content may be incorrect.

**4. Disaster Rescue Filter Applied**



**5. Reset to Show All Data**

A screenshot of a computer

AI-generated content may be incorrect.

**Tools and Technologies Used**

| **Tool** | **Purpose** |
| --- | --- |
| **Python** | Core programming language used for development |
| **Dash (Plotly)** | Framework for building interactive web applications |
| **MongoDB** | NoSQL database used to store and retrieve animal records |
| **Pandas** | Data manipulation and processing |
| **Plotly Express** | Charting and data visualization |
| **Dash Leaflet** | Geolocation visualization using maps |

**Why MongoDB?**

MongoDB was chosen as the database because:

* It provides flexible schema suitable for JSON-like data storage.
* It integrates well with Python using PyMongo.
* It allows for **efficient queries** and indexing, which is useful for filtering data dynamically.

**Why Dash?**

Dash provides a robust framework for building data-driven dashboards:

* Modular Components: Supports HTML, DataTables, Graphs, and Maps.
* Interactivity: Allows users to filter and interact with data dynamically.
* Seamless Integration: Works well with Pandas and Plotly for easy data visualization.

**Steps to Reproduce the Project**

**1. Install Dependencies**

Ensure you have Python installed and install required libraries:

pip install jupyter-dash dash pandas plotly pymongo dash-leaflet

**2. Set Up MongoDB**

* Use the provided animal\_shelter.py CRUD module.
* Update credentials (username, password, host, port, database, collection).

**3. Run the Dashboard**

Execute the Jupyter Notebook file ProjectTwoDashboard.ipynb:

jupyter notebook ProjectTwoDashboard.ipynb

**4. Interact with the Dashboard**

* Open the dashboard in your web browser.
* Apply filters to view different categories of rescue dogs.
* Explore data in the table, charts, and geolocation map.

**Challenges and Solutions**

**Challenge 1: MongoDB Connection Issues**

* **Issue:** Authentication errors when connecting to MongoDB.
* **Solution:** Verified credentials, used correct host and port, and ensured MongoDB server was running.

**Challenge 2: Dash Image Not Loading**

* **Issue:** The local image file (grazioso\_logo.png) was not displaying.
* **Solution:**
  + Stored the image in the same folder as the script.
  + Used Base64 Encoding to show logo.

**Challenge 3: DataTable Not Updating**

* **Issue:** The DataTable was not dynamically refreshing with new filters.
* **Solution:**
  + Ensured the callback function used db.read(query) to retrieve filtered data.
  + Removed the \_id column to prevent data format issues.

**References and Resources**

* [MongoDB Documentation](https://www.mongodb.com/docs/)
* [Dash User Guide](https://dash.plotly.com/)
* [Plotly Express](https://plotly.com/python/plotly-express/)
* [Dash Leaflet for Geolocation](https://dash-leaflet.herokuapp.com/)

**Developed for Grazioso Salvare | CS-340 Project | Jeffrey Flores**