Grazioso Salvare Dashboard

Lead Developer: Eric Foster

Project Overview

This project implements a full-stack dashboard for Grazioso Salvare, an international rescue animal training organization. The dashboard allows users to interact with and visualize data from the Austin Animal Center Outcomes dataset to identify dogs suitable for different types of search-and-rescue training.

The application uses a MongoDB database as the backend and a Python Dash framework for the front-end dashboard, following the MVC design pattern:

- **Model:** MongoDB database containing shelter dog data.
- **View:** Dash dashboard components (data table, charts, interactive options).
- **Controller:** Python CRUD module that manages database queries based on user input.

The dashboard is open-source and fully accessible for adaptation by similar organizations.

Features / Functionality

The dashboard includes the following key features:

1. **Interactive Data Table**
    - Displays all Austin Animal Center Outcomes data by default.
    - Pagination, sorting, and filtering capabilities are available to improve usability.
2. **Filter Options for Rescue Type**
    - Users can filter dogs by:
        - Water Rescue
        - Mountain or Wilderness Rescue
        - Disaster or Individual Tracking
    - Reset option returns all data to the original unfiltered state.
3. **Dynamic Charts**

    - **Geolocation chart:** Displays the locations of shelter animals dynamically based on applied filters.
    - **Secondary chart:** Visualizes another aspect of the dataset (breed distribution).
4. **Branding and Identification**
    - Grazioso Salvare logo displayed on the dashboard.
    - Unique identifier included: Created By: Eric Foster
5. **Responsive Dashboard Updates**
    - All charts and data tables update dynamically in response to filter selection using the controller logic.

Tools and Technologies

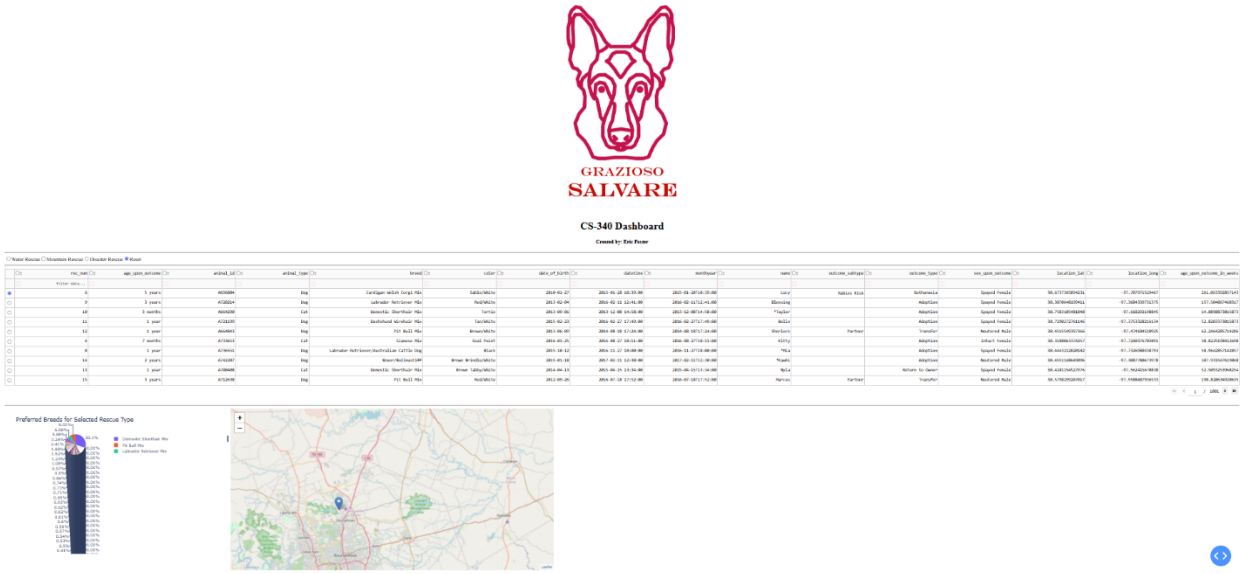
- **MongoDB:**
    - Chosen for its flexibility and ease of integration with Python.
    - Supports CRUD operations through a Python module and can efficiently store JSON-like documents representing dog profiles.
- **Python Dash Framework:**
    - Provides a structure for creating interactive, web-based dashboards.
    - Enables the use of **MVC architecture**, where the model interacts with the database, the controller handles queries, and the view displays dynamic content.
- **Plotly Express:**
    - Used for dynamic and interactive visualizations, including geolocation mapping and other charts.
- **Jupyter Notebook / IPython:**
    - Used as the environment to develop and test the dashboard.

Project Setup / Reproduction Instructions

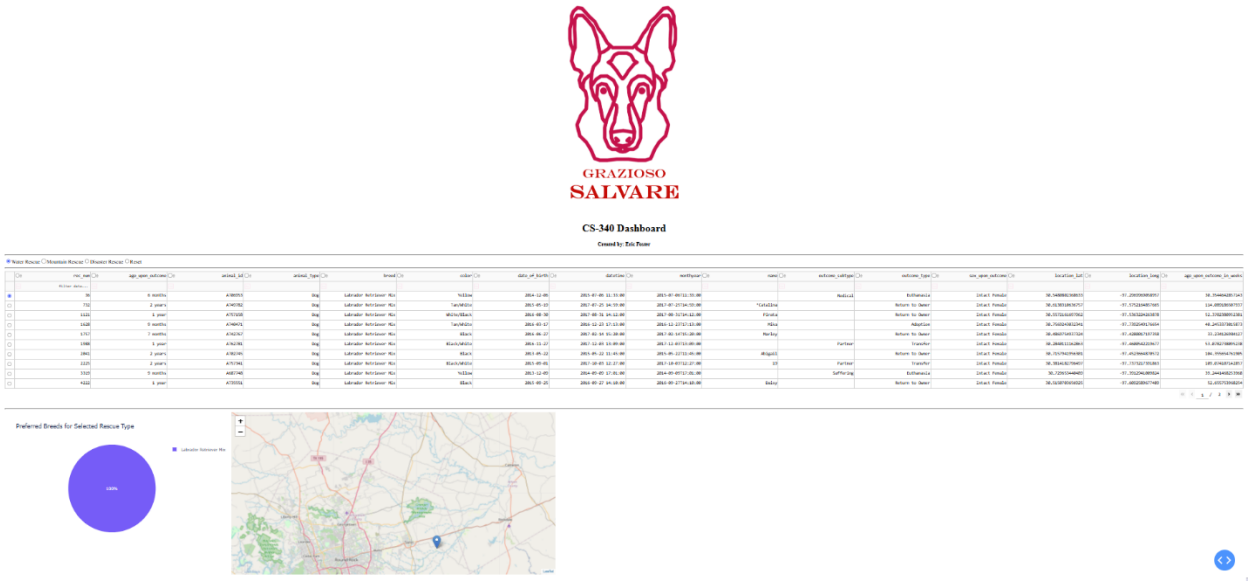
1. **Clone the repository**
    - git clone <your repo url>
    - cd <your folder name>
2. **Install dependencies**
    - pip install -r requirements.txt
3. **Database Setup**
    - View the Python CRUD README for more info on setting up the database environment.
4. **Run the Dashboard**
    - Update the username and password in the ProjectTwoDashboard.ipynb that is used to access your database
    - jupyter notebook ProjectTwoDashboard.ipynb
        1. Open the notebook and run all cells.
        2. The dashboard will launch with the data table, filter options, and charts.
5. **Using Filters**
    - Apply rescue type filters via the radio buttons.
    - Observe the dynamic updates in the data table and charts.
    - Reset to view all dogs.

Screenshots / Proof of Functionality

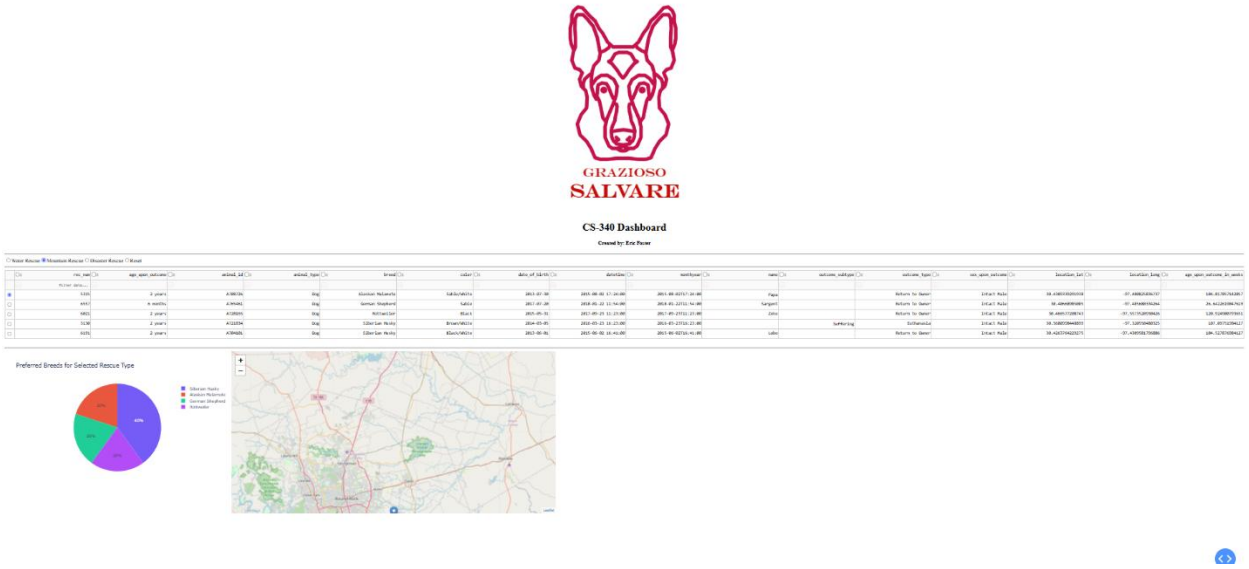
- **Initial Dashboard:**



- **Water Rescue Filter Applied:**



- **Mountain Rescue Filter Applied:**



- **Disaster Tracking Filter Applied:**

Challenges and Solutions

1. **Dynamic filtering across multiple widgets:**
    - **Challenge:** Ensuring data table and charts update simultaneously.
    - **Solution:** Implemented a controller function in the CRUD Python module that is called by Dash callback functions.
2. **Integrating geolocation data into visualizations:**

- **Challenge:** Displaying accurate locations of shelters and animals.
- **Solution:** Used Plotly Express scatter maps with proper latitude/longitude mapping.

3. **Managing large datasets in the dashboard:**
   - **Challenge:** Displaying hundreds of rows efficiently.
   - **Solution:** Enabled pagination and limited rows in the interactive table.

Resources

- MongoDB Documentation
- Dash by Plotly Documentation
- Plotly Express Documentation
- Jupyter Notebook Documentation