# Grazioso Salvare Dashboard Project README

## About the Project

This project is a web application dashboard developed for Grazioso Salvare, an organization specializing in training rescue dogs. The dashboard connects to a MongoDB database and provides an intuitive interface to filter, view, and analyze data about animal outcomes from Austin Animal Center. Users can apply filters, view detailed data in an interactive table, and analyze the results with geolocation and breed distribution charts.

**Required Functionality**

1. **Data Table**: Displays the full dataset of animals from the MongoDB database and updates dynamically based on user-selected filters.
2. **Filters**: Users can filter data by:

* Water Rescue
* Mountain or Wilderness Rescue
* Disaster or Individual Tracking
* Reset (show all data)

1. **Charts**:

* A geolocation map showing the filtered animal data.
* A pie chart displaying the breed distribution of filtered data.

1. **Branding**: The dashboard includes the Grazioso Salvare logo and a unique identifier (developer name).

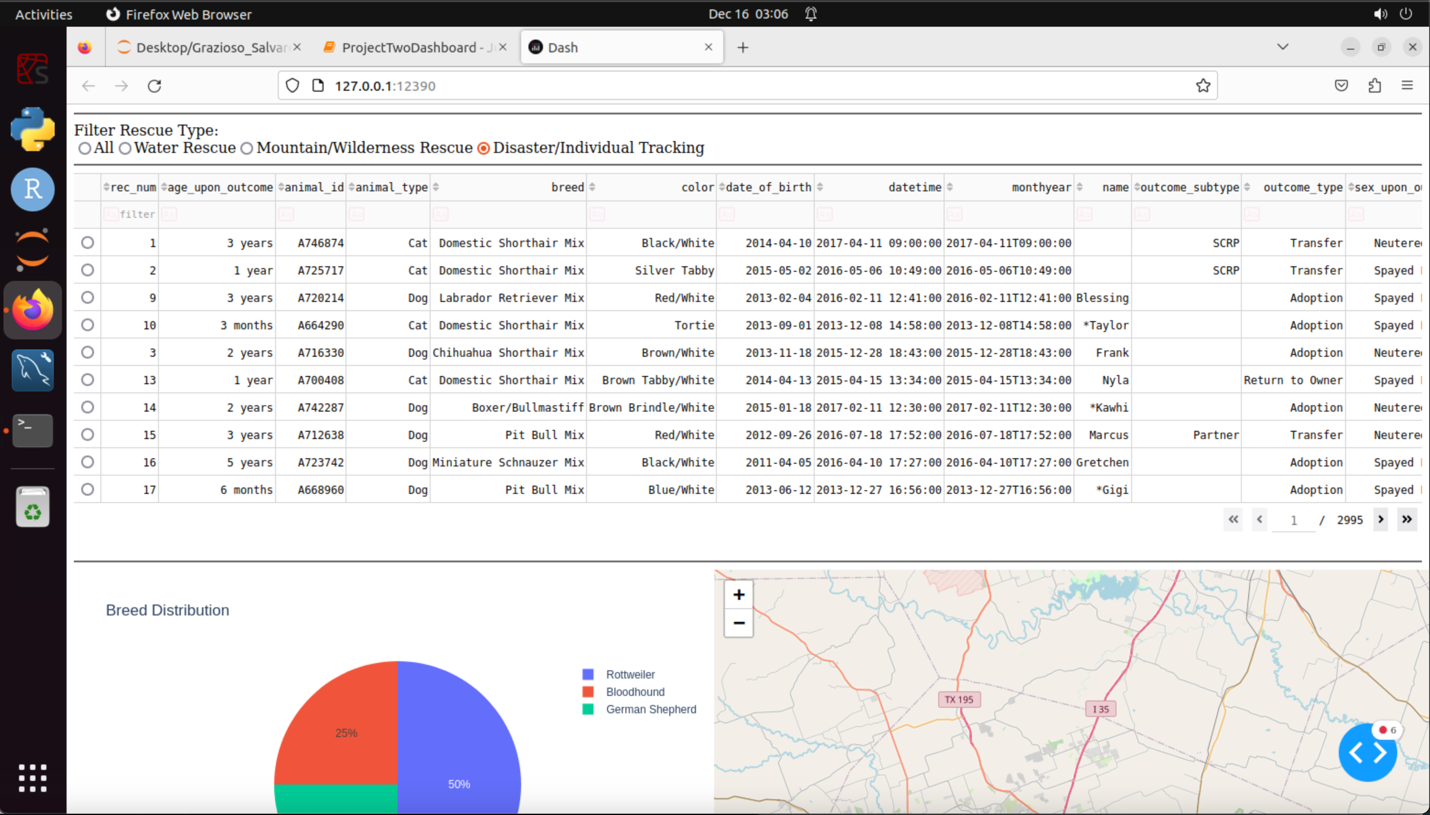
**Screenshots**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

**Tools and Rationale**

1. **MongoDB**:

* Chosen as the model component for its flexibility and ease of integration with Python.
* Stores data as documents, which are easily queried and manipulated using PyMongo.
* Provides robust performance for CRUD operations and filtering large datasets.

1. **Dash Framework**:

* Used to develop the dashboard for its simplicity and ability to create interactive web applications in Python.
* Provides components like dash\_table.DataTable, dcc.Graph, and dash\_leaflet.Map for creating dynamic and intuitive visualizations.
* The MVC (Model-View-Controller) design pattern in Dash ensures clean separation of concerns:
  + **Model**: MongoDB queries handled by the CRUD module.
  + **View**: Dash components for user interaction and display.
  + **Controller**: Dash callbacks to handle dynamic updates.

1. **Python**:

* Core programming language for implementing MongoDB queries, data manipulation, and creating the dashboard.

**Steps to Reproduce**

1. **Database Setup**:

* Ensure MongoDB is running with the aac\_shelter\_outcomes.csv dataset imported into the AAC database and the animals collection.
* Verify the connection using the following command:

mongosh --host nv-desktop-services.apporto.com --port 33257 -u aacuser -p CS340module3

1. **Environment Setup**:

* Install necessary Python libraries:

pip install pymongo dash dash-leaflet pandas plotly

1. **Run the Dashboard:**

* Navigate to the project directory and run the ProjectTwoDashboard.ipynb file using Jupyter Notebook or Jupyter Lab.
* Ensure the logo file (Grazioso Salvare Logo.png) is in the project directory.

1. **Using the Dashboard**:

* Open the dashboard in a browser (the local server URL will be displayed in the terminal).
* Apply filters using the radio buttons to view data for specific rescue types.
* Observe the interactive updates in the data table, geolocation map, and pie chart.

**Challenges and Solutions**

1. **Callback Errors**:

* Initial errors in Dash callbacks were resolved by adding debugging print statements to verify data flow and ensuring data formats were compatible with Dash components.

1. **Logo File Missing**:

* Corrected the file path to ensure the Grazioso Salvare logo displayed correctly on the dashboard.

1. **Large Dataset Handling**:

* Optimized filtering logic in MongoDB queries to ensure efficient retrieval and display of data.

1. **Serialization Issues**:

* Removed the \_id field from MongoDB documents to avoid JSON serialization errors when converting to a Pandas DataFrame.

## Contact

Aeriel Denmark