**Project Overview**

This is a full-stack web application dashboard developed for Grazioso Salvare, an international rescue-animal training company. The dashboard allows users to identify and categorize dogs from animal shelters that are suitable candidates for search-and-rescue training based on specific criteria.

The application interfaces with a MongoDB database containing animal shelter data from five shelters around Austin, Texas. Users can filter dogs based on specific rescue types (Water Rescue, Mountain/Wilderness Rescue, or Disaster/Individual Tracking) and view relevant information through an interactive data table and visualizations.

**Functionality**

Required Functionality:

1. Interactive Filtering Options: Users can select different rescue types with buttons to filter the database.
2. Dynamic Data Table: Display animal data that responds to the filtering options, with pagination, sorting, and filtering capabilities.
3. Geolocation Chart: Shows the locations of animals on a map that updates based on filter selections.
4. Breed Distribution Chart: A pie chart showing the distribution of dog breeds in the current filtered data set.
5. Branding: Includes the Grazioso Salvare logo and developer identification.

**Screenshots of Functionality**

Starting State:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a map

AI-generated content may be incorrect.

Water Rescue Filter Applied:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

Mountain or Wilderness Rescue Filter Applied:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a map

AI-generated content may be incorrect.

Disaster or Individual Tracking Filter Applied:

A screenshot of a computer screen

AI-generated content may be incorrect.

A map of a city

AI-generated content may be incorrect.

Reset State (Return to Unfiltered View):

A screenshot of a computer screen

AI-generated content may be incorrect.

A map with a location pin

AI-generated content may be incorrect.

**Tools and Technologies**

MongoDB

MongoDB was chosen as the database for this project for several reasons:

* Document-oriented Structure: The flexible schema allows for sorting complex animal data in a way that matches the application’s needs.
* Query Capabilities: MongoDB’s query language is powerful for filtering animals based on multiple criteria.
* Python Integration: MongoDB integrates seamlessly with Python through the PyMongo driver, making it ideal for this stack
* Scalability: As the animal shelter data grows, MongoDB can scale to accommodate larger datasets.

Python

Python serves as the backend programming language for this project:

* PyMongo: Used to create the CRUD module that interfaces with the MongoDB database.
* Pandas: Helps with data manipulation and preparation for visualization.
* Dash Framework: Provides the view and controller structure for the web application.

Dash Framework

Dash is a Python framework for building analytical web applications:

* Component Libraries: Dash provides core components (dcc) and HTML components for building the user interface.
* Interactive Visualizations: Dash integrates with Plotly for creating interactive charts and maps.
* Callbacks: Dash’s callback functions enable the interactive nature of the dashboard, connecting user inputs to visual outputs.
* MVC Pattern Support: Dash enables the Model-View-Controller design pattern, where MongoDB is the model, dashboard widgets are the views, and Python callbacks are the controllers.

**Project Completion Steps**

1. Database Setup:
   * Created a MongoDB database to store the animal shelter data.
   * Imported the Austin Animal Center Outcomes dataset.
2. CRUD Module Development:
   * Developed a Python module (AnimalShelter.py) with Create, Read, Update, and Delete functionality.
   * Implemented error handling and validation for database operations.
3. Dashboard Development:
   * Created the basic dashboard layout with Dash, including the logo and identifier.
   * Implemented the unfiltered data table to display all animals.
   * Developed database queries for filtering dogs based on rescue types.
   * Created interactive radio buttons for selecting filter options.
   * Implemented the geolocation chart using Plotly Scattermapbox.
   * Added a pie chart to visualize breed distribution.
   * Connected all components through Dash callbacks for interactive functionality.
4. Testing and Deployment:
   * Tested all dashboard functionality to ensure proper interactions.
   * Captured screenshots of the dashboard in various filtering states.
   * Documented the project in this README file.

**Challenges and Solutions**

1. Challenge: MongoDB Data Integration
   * Solution: Created a robust CRUD module with error handling to ensure reliable database operations. Used the PyMongo driver to interface between Python and MongoDB.
2. Challenge: Dynamic Data Visualization
   * Solution: Implemented Dash callbacks to update visualizations in real-time based on user selection. Used Plotly's mapping and charting capabilities for interactive visualizations.
3. Challenge: Filtering Logic Implementation
   * Solution: Carefully structured MongoDB queries to match the specific criteria for each rescue type, considering breed, age, and other factors as specified by Grazioso Salvare.
4. Challenge: User Interface Design
   * Solution: Created an intuitive interface with clear filtering options and responsive visualizations that update automatically. Added features like pagination, sorting, and filtering to enhance user experience.

**Resources and References**

* MongoDB Documentation
* PyMongo Documentation
* Dash Documentation
* Plotly Documentation
* Austin Animal Center Dataset

Installation and Setup Instructions

1. Prerequisites:
   * Python 3.6+
   * MongoDB
   * Required Python packages: pymongo, dash, pandas, plotly
2. Database Setup:
   * Install MongoDB Community Edition
   * Create a database named "AAC"
   * Create a user "aacuser" with password access
   * Import the Austin Animal Center Outcomes dataset
3. Application Setup:
   * Clone this repository
   * Install required packages: pip install -r requirements.txt
   * Ensure the Grazioso Salvare logo is in the project directory
   * Run the dashboard: python -m jupyter notebook ProjectTwoDashboard.ipynb
4. Access the Dashboard:
   * After running the notebook, execute all cells
   * The dashboard will be accessible at http://localhost:8050