README – Grazioso Salvare Search & Rescue Dashboard

1. PROJECT OVERVIEW

This project implements a fully functional MongoDB-based Search and Rescue Dashboard for Grazioso Salvare, an international animal training organization. The purpose of this dashboard is to assist the company in identifying animals that are the best candidates for specialized search and rescue missions such as water rescue, mountain/wilderness rescue, and disaster/individual tracking.

The application connects directly to a MongoDB database that stores real shelter animal data provided by the Austin Animal Center. It retrieves, filters, and visualizes these records through an interactive dashboard built with Python, Dash, and Plotly, using a Model-View-Controller (MVC) architecture pattern.

The completed dashboard allows users to:

-View an interactive data table of animals from the MongoDB database.

-Apply filter options based on three specific rescue categories.

-View dynamic visualizations, including:

-A breed distribution bar chart (top 10 breeds in view).

-An interactive geolocation map that updates based on the selected animal.

-Export current filtered data to CSV for external analysis.

-Navigate an intuitive, branded interface with the official Grazioso Salvare logo and SNHU course credits.

2. REQUIRED FUNCTIONALITY (SCREENSHOTS AND PROOF)

The dashboard meets and exceeds all rubric requirements for interactivity, design, and functionality.

Interactive Features:

-A radio button selection menu for Water Rescue, Mountain/Wilderness Rescue, Disaster/Individual Tracking, and Reset (All).

-Each selection updates the database query dynamically, modifying the animal table, breed distribution chart, and map marker simultaneously.

-Selecting a single animal row displays its location on a live interactive map centered over Austin, Texas.

Proof of Functionality (Screenshots and Video):

1. 1 reset all view.png – Default unfiltered dataset view

2. 2 water rescue.png – Filter applied for water rescue candidates

3. 3 mountain wilderness.png – Filter applied for mountain/wilderness rescues

4. 4 disaster rescue.png – Filter applied for disaster/individual tracking rescues

5. filters.mp4 – Demonstration video showing selection of each animal and dynamic map updates

All components perform dynamically and without error

3. TOOLS AND TECHNOLOGIES USED

-MongoDB – Non-relational database used to store and query animal records efficiently

-Python – Core programming language integrating data retrieval, transformation, and display logic

-Pandas – Converts MongoDB query results into DataFrames for table and chart rendering

-Dash by Plotly – Framework for building interactive dashboards with live updates

-dash\_table.DataTable – Provides sortable, searchable, and exportable data tables

-dash\_leaflet – Plots geolocation markers for selected animal coordinates

-Plotly Express – Generates dynamic and interactive bar charts

-JupyterDash – Integrates Dash apps directly within JupyterLab for development and testing

These tools were selected for their modular structure, flexibility, and compatibility with SNHU’s learning environment

4. DEVELOPMENT PROCESS

This project followed an MVC inspired pattern:

-Model: CRUD\_Python\_Module.py handled all database communication (CRUD operations)

-View: Dash and HTML components defined the user interface

-Controller: Dash callbacks acted as the logic layer, handling filter changes and data updates

Development Steps:

1. Established MongoDB connection in the CRUD module

2. Created build\_query() function to define filters for each rescue type

3. Verified data retrieval into Pandas DataFrames

4. Built main layout (logo, header, filter panel, data table, chart, and map)

5. Implemented callbacks for updating table, chart, and map

6. Styled interface with consistent layout and branding

7. Tested all filters, verified accuracy, and captured proof screenshots and video

5. CHALLENGES AND SOLUTIONS

-Challenge: IndentationError – “unindent does not match any outer indentation level.”

-Solution: Standardized all indentation to 4 spaces in callbacks and exception blocks

-Challenge: Chart error due to missing breed column

-Solution: Normalized DataFrame and ensured breed column populated before plotting

-Challenge: Map not centering when no selection made

-Solution: Added fallback coordinates to center on Austin, Texas

-Challenge: Layout inconsistencies between browsers

-Solution: Applied flexible “display: flex” containers and width constraints

-Challenge: Table overflow due to long data fields

-Solution: Added horizontal scrolling (overflowX) and enabled pagination

Each issue was debugged, corrected, and retested until all components worked

6. REPRODUCTION INSTRUCTIONS

To reproduce and run this project locally:

Step 1: Install Dependencies

pip install dash dash\_leaflet pandas numpy plotly jupyter-dash pymongo

Step 2: Ensure MongoDB is Running

Import or verify the Austin Animal Center dataset is loaded into your MongoDB instance.

Default credentials: username = "aacuser", password = "SNHU1234"

Step 3: Open the Project

Open ProjectTwoDashboard.ipynb in JupyterLab.

Step 4: Run All Cells

Ensure CRUD\_Python\_Module.py and dashboard execute without errors.

Step 5: Start the Server

app.run\_server(mode="jupyterlab")

Step 6: Interact with Dashboard

Use radio buttons to test all filters.

Select individual animals to view map markers update dynamically.

Step 7: Export Data

Use “Export” at the top of the table to save current results as CSV.

7. DEMONSTRATION MEDIA

1 reset all view.png – Default unfiltered data view

2 water rescue.png – Water rescue filter applied

3 mountain wilderness.png – Mountain/Wilderness rescue applied

4 disaster rescue.png – Disaster/Individual Tracking applied

filters.mp4 – Demonstration video showing interactive filter selections and live map movement

All images and video are stored in the “/screenshots” directory within the submission ZIP

8. REFERENCES

Austin Animal Center Outcomes Dataset (2020). City of Austin, Texas Open Data Portal.

https://doi.org/10.26000/025.000001

Dash Documentation: https://dash.plotly.com/

dash\_leaflet Documentation: https://dash-leaflet.herokuapp.com/

MongoDB Documentation: https://www.mongodb.com/docs/

Southern New Hampshire University (SNHU) CS-340 Course Materials

9. CONCLUSION

This project demonstrates the successful creation of a complete client/server dashboard application integrating MongoDB with Python and Dash. It fulfills Grazioso Salvare’s business goal of identifying ideal animal candidates for rescue training quickly and visually

The dashboard’s interactivity, data filtering, and live mapping provide clear analytical insight while maintaining strong usability and performance