Erek Mackroy

CS-340 Client/ Server Development

Project Two

For this project, I created an interactive dashboard application for Grazioso Salvare using Python, MongoDB, and Dash. The goal was to design a visual tool that allows users to explore and analyze animal shelter data from MongoDB and filter it based on rescue types such as Water Rescue, Mountain or Wilderness, and Disaster or Individual Tracking. This dashboard gives the organization an easier way to identify potential rescue animals based on their training and other attributes. The dashboard connects directly to the database, retrieves live data, and updates automatically when filters are applied. It includes a table for browsing animal information, a bar chart showing top breeds, and a map with location markers. I also included the Grazioso Salvare logo on both sides of the title, which keeps the design consistent with the company’s branding requirements.

The main purpose of the dashboard is to help Grazioso Salvare make better use of their data. Instead of manually searching through large spreadsheets or raw database records, users can quickly apply filters to narrow down their search to animals suitable for specific rescue missions. The dashboard provides a clear and organized interface where all data, visualizations, and location details appear in one place. It includes a filter panel on the left where users can choose between Reset (All Records), Water Rescue, Mountain or Wilderness, and Disaster or Individual Tracking. The dynamic data table instantly updates to reflect those filters and displays key information such as the animal’s ID, type, breed, and color. The bar chart shows the top breeds currently visible in the table, and the map pinpoints the location of selected animals, allowing users to see both the data and geography at the same time.

The database connection is handled through a custom Python module called CRUD\_Python\_Module.py, which performs the necessary database interactions. It includes the standard CRUD operations—Create, Read, Update, and Delete—but for this project, the Read function is the most important. The dashboard uses it to retrieve live data from MongoDB depending on which radio button is selected. For example, if the Water Rescue filter is chosen, only animals trained for water-related rescues are shown. If Mountain or Wilderness is selected, the query retrieves animals trained for rugged terrain rescues, while Disaster or Individual Tracking displays those prepared for search and recovery or tracking operations. This system ensures each dataset returned to the user is accurate and relevant to their chosen category.

The dashboard itself uses Dash, Plotly, and Dash Leaflet to combine interactivity with visual clarity. The data table displays all the information in an easy-to-read format, the bar chart summarizes breed counts for quick insight, and the map adds a spatial layer of understanding. The dashboard layout was designed to look professional and balanced, with the Grazioso Salvare logo displayed above and beside the “CS-340 Dashboard” title. I resized the logos slightly to make them visible but not distracting, keeping the focus on the dashboard content. These design choices make the dashboard user-friendly and professional enough for real use.

I tested the dashboard to confirm that data updates correctly when filters are switched, that the charts change dynamically, and that the map marker correctly updates with each animal selection. The main issues I ran into early on involved getting the data to load properly and adjusting image sizes in the table. Once those were corrected, everything worked smoothly. The dashboard now loads data instantly, changes filters without delay, and keeps all components synchronized.

To run the project, MongoDB needs to be started with the animal dataset loaded. After that, JupyterLab can be opened, and the file ProjectTwoDashboard.ipynb should be run cell by cell. When the app launches, the link provided (for example, <http://127.0.0.1:8051/>) can be opened in a browser to view the live dashboard. The interface will automatically show all records by default, and users can begin exploring by selecting any of the filters on the left-hand side.

All data used in this project is educational and does not include any sensitive or private information. The system only performs read operations, so no data can be altered or deleted from the database. This ensures ethical use and protects the integrity of the dataset while still allowing meaningful exploration.

Working on this project gave me practical experience in combining database functionality with front-end visualization. It helped me better understand how data can move seamlessly between a backend server and an interactive user interface. I learned how to handle callbacks in Dash, manage real-time updates, and create an experience that’s visually clean and functional. If I continue this project, I’d like to add user authentication for secure access, expand visualization options, and include features that let users export reports for different rescue types.

Overall, this project successfully demonstrates how databases and dashboards can come together to solve a real-world problem in a professional setting. The result is a clean, fully functional dashboard that meets all the goals outlined in the CS-340 Project Two guidelines and rubric.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.