**About**

This interactive dashboard was developed for Grazioso Salvare, a company that identifies dogs that are good candidates for search-and-rescue training. The dashboard integrates with a MongoDB database of animal records provided by the Austin Animal Center. It is tailored specifically to meet Grazioso Salvare’s needs by allowing users to filter the dogs by type, including breeds ideal for water, mountain, wilderness, and disaster rescue, as well as individual tracking. These results are displayed in a pie chart, allowing users to quickly and easily identify which specific breeds are available in these categories. A geolocation feature provides quick access to a specific animal’s location within the Austin, TX, area.

**Requirements**

* Python 3.x
* JupyterDash
* Dash
* DashLeaflet
* Plotly
* Pandas
* NumPy
* PyMongo
* PyCRUD4Mongo

**Installation**

1. Install required packages  
   
2. Clone/download the repository  
   
3. Start MongoDB  
   
4. Load the dataset  
   
5. Run the dashboard  
   
6. Execute the script  
   A screenshot of a computer program

   AI-generated content may be incorrect.

**Screenshots**

A screenshot of a map

AI-generated content may be incorrect.

A screenshot of a map

AI-generated content may be incorrect.

A screenshot of a map

AI-generated content may be incorrect.

A screenshot of a map

AI-generated content may be incorrect.

**Technical**

MongoDB was the database of choice for this project, as it provides a flexible, document-oriented schema that does not force rigid tables. This was beneficial as the animal records provided varied in their content, with some including certain information while others omitted it. Python is an excellent language for processing and displaying this data, as well as having a large supporting suite of modules. The PyMongo module allows us to easily hook into the MongoDB database and construct an API (PyCRUD4Mongo) for ease of access to this data through CRUD operations. The Dash module provided the View and Controller layers of the MVC architecture, with the View being defined by Dash’s HTML and core components, while the Controller was implemented through callback functions, which seamlessly update the dashboard when it is interacted with. The JupyterDash and DashLeaflet modules further extended these capabilities, with JupyterDash allowing the system to be executed from within Jupyter Lab and DashLeaflet providing the geolocating map and markers.

**Development**

This project was developed in three major steps. The first step was development of the CRUD wrapper. This was essential as it simplifies accessing data from within the MongoDB database. The next step was developing a general dashboard. This provided valuable information and experience working with Dash and how to implement specific features within it. The final step was developing the final dashboard with the specific features required by Grazioso Salvare, such as filtering by specific queries and providing a chart for easily visualizing the data.