# CS 340 README Joel Hays

## Project Overview

The Grazioso Salvare Dashboard is a user-friendly web application designed to help the Grazioso Salvare organization identify and categorize dogs suitable for search-and-rescue training within the Austin, Texas region. The dashboard interfaces with a MongoDB database to dynamically display and filter data, providing valuable insights through various interactive widgets and charts.

## Required Functionality

The dashboard includes the following functionalities:

1. **Interactive Filters**: Options to filter data based on rescue types:
   * Water Rescue
   * Mountain or Wilderness Rescue
   * Disaster or Individual Tracking
   * Reset (to view all data)
2. **Dynamic Data Table**: Displays relevant information such as animal ID, breed, color, age, and location, and updates based on selected filters.
3. **Geolocation Chart**: Shows the exact location of the selected dog within the Austin, Texas region.
4. **Bar Chart**: Displays the ages of dogs by breed, updating dynamically based on selected filters.
5. **Branding**: Includes the Grazioso Salvare logo\*\* and a unique identifier with the developer's name.

Non-verbal Video Demonstration Link:

<https://www.loom.com/share/9f739d0130484ae08e4704fd96b9bbd0?sid=9b42e3c3-661c-4ad2-8650-12afb2f03623>

Tools used:

* [MongoDB](https://www.mongodb.com/): Used due to its ease of scale, flexibility and seamless integration for use with datasets.
* [Dash Framework](https://dash.plotly.com/):
  + Allows for highly interactive web applications with minimal code
  + Integration with plotly allows the use of different visualization tools like graphs and charts
  + Ease of use allows more focus on content rather than how to make it work
* [PyMongo](https://pymongo.readthedocs.io/en/stable/): Allows integration between python coding language and MongoDB(NoSQL)
* [Jupyter Notebook](https://docs.jupyter.org/en/latest/): An open-source web application that allows you to create and share documents that contain live code. Used for the interactive environment, visualization support, and easy documentation.

Project Implementation Steps

1. **Set Up MongoDB**:
   * Installed and configured MongoDB locally.
   * Created a database and collection to store the animal shelter data.
   * Loaded the Austin Animal Center Outcomes dataset into the MongoDB collection.
2. **Develop CRUD Module(Project 1)**:
   * Implemented a Python module for CRUD operations using PyMongo.
   * Ensured the module could perform create, read, update, and delete operations on the MongoDB collection.
3. **Design Dashboard Layout**:
   * Set up the initial layout of the dashboard using Dash.
   * Created components for the data table, geolocation chart, and bar chart.
   * Included the Grazioso Salvare logo\*\* and unique identifier.
4. **Implement Interactive Filters**:
   * Added dropdowns for filtering data based on rescue types.
   * Developed callback functions to update the data table, geolocation chart, and bar chart based on selected filters.
5. **Test and Deploy Dashboard**:
   * Ran the dashboard locally to ensure all components and filters functioned correctly.
   * Captured screencast of the dashboard for database demonstration.

Challenges and Conclusion:

This Project was interesting with the integration of Python and NoSQL. This was also the first time I have been exposed to a Linux based OS. Newer challenges were incorporating the logo into the finished product. I was having trouble finding its location and mapping it in the Jupyter portal. Furthermore, I was having trouble with sorting because there are many mixed breeds in the database. This caused me to find a way to allow for anything containing any of the breeds required using the $regex query.

*Note\*\* The logo icon is accessed via a created assets file in Jupyter Notebook*

## Contact

Your name: Joel Hays; joel.hays@snhu.edu