**CS 340 Project 2 README**

**About the Project**

This project is the second phase of a full-stack web application designed to help Grazioso Salvare, an international organization that trains dogs for search-and-rescue missions. The goal is to build an easy-to-use client-facing dashboard using Dash, which allows users to visually interact with animal shelter data stored in a MongoDB database. The dashboard helps Grazioso Salvare quickly identify and categorize dogs that are best suited for different types of rescue work, like water, mountain or wilderness, and disaster or individual tracking, based on preferred factors like breed, sex, and age. Currently, this process is mostly manual and takes a lot of time. By automating the filtering of data from the shelters, the dashboard makes it faster to identify ideal candidates for training. This efficiency not only helps Grazioso Salvare respond more effectively to emergencies but also allows other rescue organizations to adopt and benefit from the tool.

**Screenshots**

Starting state of dashboard:

*Dashboard displays all records from the MongoDB database.*

A screenshot of a computer screen

AI-generated content may be incorrect.

Water Rescue filter applied:

*Shows dogs that match the preferred criteria for water rescue.*

A screenshot of a computer

AI-generated content may be incorrect.

Mountain or Wilderness Rescue filter applied:

*Shows dogs that match the preferred criteria for mountain or wilderness rescue.*

A screenshot of a computer screen

AI-generated content may be incorrect.

Disaster or Individual Tracking filter applied:

*Shows dogs that match the preferred criteria for* disaster or individual tracking.

A screenshot of a map

AI-generated content may be incorrect.

Reset filters:

*Returns the dashboard to its original and unfiltered state.*

A screenshot of a computer screen

AI-generated content may be incorrect.

**Installation**

* MongoDB is used to store and manage animal shelter data. MongoDB is a NoSQL database that supports flexible schemas and can efficiently handle large collections of documents, making it ideal for this project, and works well with Python.

Download: <https://www.mongodb.com/try/download/community>

* Python 3 is the main programming language used for building both the CRUD module and the dashboard. Python’s readability and extensive library support make it well-suited for data and web projects.

Download: <https://www.python.org/downloads/>

* Dash is used to create interactive web applications, and it provides all the components needed to build the dashboard interface. In this project, I used Dash to create the layout and to control how the dashboard responds when a user interacts with it.

Install: <https://dash.plotly.com/installation>

* Jupyter Notebook is used to test and develop both the CRUD module and the dashboard in an interactive coding environment. It allows for easy debugging and visualization of code execution.

Install: <https://jupyter.org/install>

* PyMongo lets Python communicate with the MongoDB database.

Install: <https://pypi.org/project/pymongo/>

* JupyterDash lets you run Dash apps inside Jupyter Notebook. This was used to build the dashboard interface for the project without needing a separate web server.

Install: <https://dash.plotly.com/dash-in-jupyter>

* Dash Leaflet enables mapping features within the Dash dashboard. It is used to display animal locations on an interactive map.

Install: <https://www.dash-leaflet.com/>

* Pandas is used to structure and filter the shelter data.

Install: <https://pandas.pydata.org/docs/getting_started/install.html>

* Plotly lets you create interactive charts like the pie chart in the dashboard.

Install: <https://plotly.com/python/pie-charts/>

**Getting Started**

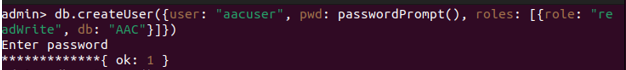
Follow these steps to copy the project and run it locally:

1. Install the required tools listed in the Installation section. You’ll need MongoDB, Python 3, Jupyter Notebook, and the required Python packages, such as pymongo, pandas, dash, dash-leaflet, plotly, and jupyter-dash.
2. Download the project files to your local machine. This includes animal\_shelter\_crud.py file, which contains the CRUD logic, the ProjectTwoDashboard.ipynb notebook file for running the dashboard, the aac\_shelter\_outcomes.csv file with the Austin Animal Center records, and the Grazioso Salvare Logo.png image for branding.
3. Set up your MongoDB database and import the AAC dataset containing all animal records.

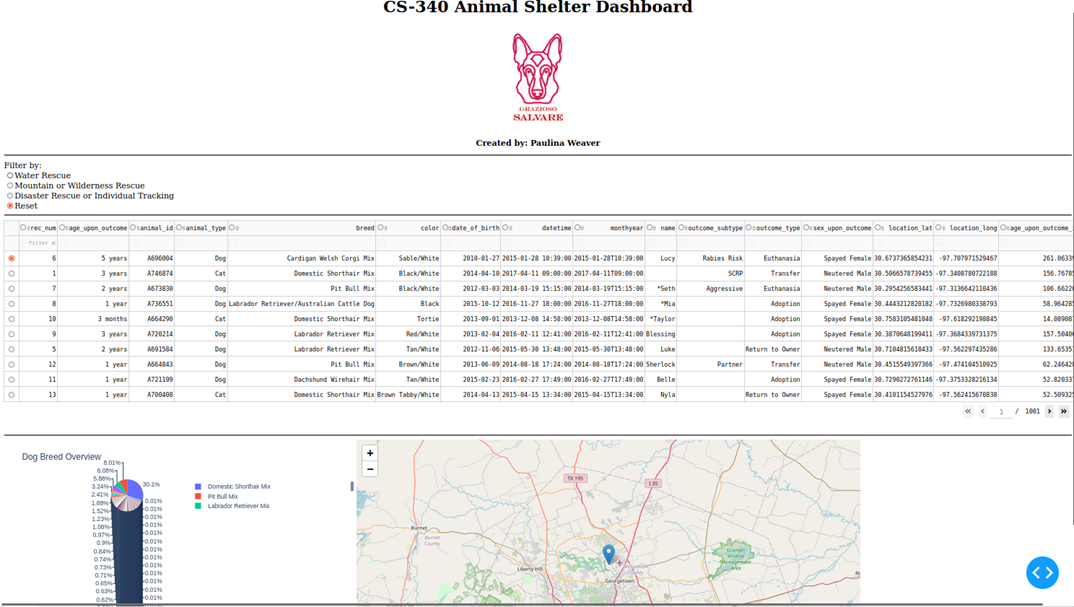
A screen shot of a computer

AI-generated content may be incorrect.

1. Create an aacuser in MongoDB with the role readWrite on the AAC database. This allows the Python code to perform operations securely.



1. Open Jupyter Notebook and launch the ProjectTwoDashboard.ipynb file. Running the cells will start the dashboard locally in your web browser. From there, you can interact with the data, apply filters, and explore the rescue dog recommendations visually.



**Challenges**

One challenge I ran into was the map not showing up when the dashboard first loaded. After some debugging and testing the code in small batches, I realized it was because the default row selection was empty at startup. Since the map depends on the latitude and longitude coordinates from the selected animal, it wouldn’t load properly without the valid data. To fix this, I added error handling and made the code default to the first row if no selection was made. I also added a message to show if location data was missing, which made the map more reliable on load.

Another thing that bothered me was that the map didn’t automatically center on each selected animal. Sometimes, the animal’s marker would even be outside the visible area so the used would have to search for the animal in the map. To improve the user experience, I updated the code so the map always centers on the location of the selected animal instead of using a fixed map position. This way, no matter which animal is clicked, the map zooms in and shows it clearly.

**Contact**

Paulina Weaver