# Final Project README

## About the Project/Project Title

**Grazioso Salvare Search-and-Rescue Dashboard:** This project aims to create a dashboard for Grazioso Salvare, a search-and-rescue dog training organization. The dashboard utilizes data from five animal shelters in the Austin, Texas region to identify and categorize dogs fit for training. The application includes a MongoDB database and a user-friendly web dashboard to visualize and interact with the data.

## Motivation

Grazioso Salvare needs a way to identify dogs for search-and-rescue training. The dashboard provides an interface for filtering and categorizing dogs based on specific criteria, such as age and rescue type. This project aims to improve the organization's ability to select and train dogs.

## Getting Started

To run the project locally, follow these steps:

1. Install the required Python libraries using pip install.
2. Set up a MongoDB database with the necessary credentials.
3. Update the username, password, and database connection details in the code.
4. Run the dashboard application using python mylib.py.
5. Access the dashboard in your web browser.

## Installation

Python

MongoDB

Jupyter Notebook for running the JupyterDash code.

## Usage

**Explain why MongoDB was used.**

* MongoDB was chosen for its flexibility, compatibility with Python, scalability, geospatial capabilities, and strong community support.

**Explain the Dash framework.**

* Dash provides a user-friendly and efficient framework for developing interactive web applications with a clear MVC structure.

### Code Example

# Import necessary libraries

from jupyter\_dash import JupyterDash

from dash import dcc, html, dash\_table

from dash.dependencies import Input, Output

import base64

import pandas as pd

import dash\_leaflet as dl

import plotly.express as px

from mylib import AnimalShelter

# Set up JupyterDash app

app = JupyterDash(\_\_name\_\_)

# Initialize AnimalShelter database connection

username = "aacuser"

password = "SNHU1234"

db = AnimalShelter(username, password, 'your-host', 31499, 'AAC', 'animals')

# Load initial data

df = pd.DataFrame.from\_records(db.read({}))

# Configure layout

app.layout = html.Div([

# Logo and dashboard title

html.A(

html.Img(src='data:image/png;base64,{}'.format(encoded\_image.decode()), id='logo-id',

style={'display': 'block', 'margin-left': 'auto', 'margin-right': 'auto'}),

href='https://www.snhu.edu'

),

html.Center(html.B(html.H1('CS-340 Dashboard'))),

# Interactive filter options

dcc.RadioItems(

id='filter-type',

options=[

{'label': 'Water Rescue', 'value': 'water'},

{'label': 'Mountain or Wilderness Rescue', 'value': 'mountain'},

{'label': 'Disaster or Individual Tracking', 'value': 'disaster'},

{'label': 'Reset', 'value': 'reset'},

],

value='reset'

),

### Tests

To run tests, make sure you are using the proper columns names for the data table, using

* print(len(df.to\_dict(orient='records')))
* print(df.columns)

### Screenshots

*A red line drawing of a dog

Description automatically generated*

**Logo with included URL anchor tag**

**to the client’s home page: www.snhu.edu**

*A screenshot of a computer

Description automatically generated*

**Data table with filter buttons at the top. Above is the unique identifier for the project.**

*A screenshot of a computer

Description automatically generated*

**Water Rescue option.**

*A screenshot of a computer

Description automatically generated*

**Mountain rescue option.**

*A screenshot of a map

Description automatically generated*

**Disaster Rescue option.**

**A screenshot of a map

Description automatically generated**

**Reset option which resets all the filters.**

## Roadmap/Features

**Steps taken to complete the project:**

1. Understanding the requirements from Grazioso Salvare.
2. Database connection and using the necessary credentials.
3. Data processing and uploading the files given.
4. Designing the dashboard layout.
5. Implementing interactive filtering.
6. Adding visual graphs and maps.
7. Testing and debugging.

**Challenges:**

* Determining the logic for filtering data based on rescue types.
* Integrating Dash with Jupyter notebooks presented minor challenges related to setup and execution.

**Sources:**

https://plotly.com/

This was used to create the pie chart shown in the screenshots above.

## Contact

Zoe Domagalski