**Nicholas Rusinski**

**Chao Ling**

**CS-340-16420**

**Oct-19-2024**

**README for Grazioso Salvare Dashboard Project**

**Project Overview**

Grazioso Salvare is an international rescue-animal training company that identifies and trains dogs for search-and-rescue missions. This dashboard allows users to filter and visualize data from local animal shelters in the Austin, Texas area to identify suitable candidates for training.

**Required Functionality**

**Filter by Rescue Type** Users can select from various rescue scenarios (Water, Mountain, Disaster) to filter available dogs based on specific criteria.

**Data Table** A dynamic table displays detailed information about each dog, including breed, age, and sex.

**Visual Analytics** A pie chart visualizes the distribution of dog breeds among the filtered data.

**Geographic Mapping** A map shows the location of selected dogs

**Samples of the working Dashboard**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Tools Used**

**Python** The main programming language used for backend logic and data manipulation.

**Dash** A web application framework built on Flask, chosen for its simplicity in creating interactive dashboards.

**Plotly** Utilized for creating rich, interactive visualizations that enhance user engagement.

**Pandas** Employed for data manipulation and analysis, essential for processing data from the MongoDB database.

**MongoDB** A NoSQL database that allows flexible data storage and efficient querying, ideal for managing diverse datasets.

**Why MongoDB?**

MongoDB provides a schema-less data model suitable for the varying profiles of rescue dogs. Its powerful querying capabilities in Python enable easy data retrieval and manipulation, making it an excellent choice for this application.

**Dash Framework Explanation**

Dash separates the view and control logic in web applications:

**View Layer** Allows for the creation of interactive components using HTML-like syntax.

-**Control Layer** Manages user input and dynamically updates the dashboard without requiring page reloads, enhancing user experience.

**Main Application Script**

Contains the Dash dashboard layout and functionality.

Handles data retrieval, filtering, and visualization.

**Breakdown**

Automatic Database Connection: The application securely stores the database credentials, allowing for seamless connections to the MongoDB database.

Simplified HTML Styling: Utilizes the Dash framework for streamlined HTML styling, enhancing the overall user interface with minimal effort.

Customizable Data Filtering and Viewing: Dash provides various options for filtering and visualizing data, making it easier for users to find relevant information quickly.

Dynamic Callback Functions: Implements callback functions to enable real-time updates across tables and charts based on user interactions, allowing for a more interactive experience.

Visual Data Representation: Leverages Plotly to create visually appealing charts and graphs, simplifying data interpretation and analysis.

**CRUD Class**

Manages database operations for the animal shelter data.

Provides methods for creating, reading, updating, and deleting entries in MongoDB.

**Breakdown**

The application allows for adding new records to the MongoDB database through the create method. This method accepts a dictionary containing the relevant data and uses MongoDB's insert\_one function to add the document. It returns the unique ID of the inserted document, ensuring that new entries are easily traceable.

The read method dynamically handles queries of all sizes and complexities. By accepting an optional query parameter, it can return a specific subset of records or all documents if no query is provided. This flexibility allows for efficient data retrieval, accommodating varying user needs—from broad searches to targeted queries based on specific criteria. The method returns the results as a list of dictionaries, making it easy to process and display in the dashboard.

The update method facilitates modifications to existing records. Users can specify which document to update using a key-value pair for the query. This method supports both single and bulk updates, allowing for flexibility depending on the use case. It returns the count of modified documents, providing feedback on the operation's success.

The delete method enables the removal of records from the database. Similar to the update function, it requires a query to identify which documents to delete. It supports both single and multiple deletions, allowing users to clean up the database efficiently. The method returns the count of deleted documents, ensuring users are informed about the outcome of their actions.

**Set Up Environment**

Install Python

Add External Libraries:

Dash

Dash Leaflet

Plotly

Dash Table

Pandas

NumPy

Matplotlib

Jupyter Dash

PyMongo

Install MongoDB

Create the database collection and set up user administration.

Data Import

Populate the database with data from local animal shelters using the Mongo shell and the mongoimport command.

Prepare Code

Add the required Python files to a project folder, along with any images needed.

Run Code

Execute the following command in the terminal: python path/to/your\_file.py

Access the Dashboard

Open the provided URL in your browser to access the dashboard.

**Challenges Encountered**

Learning MongoDB, a schema-less database, has been an enlightening experience. My previous experience was solely with SQL-style databases, so adapting to MongoDB's flexibility was a shift. I have grown to appreciate how powerful yet simple MongoDB is, and I will definitely consider it for future projects.

HTML development with the Dash framework was another first for me, as I encountered HTML code beyond the basics. The Dash framework made it easier to create an interactive dashboard, enabling a seamless user experience.

Utilizing callback functions was particularly interesting. I learned how to bind functions based on the inputs from HTML fields, allowing for dynamic updates of the dashboard elements. This feature greatly enhances user interaction and provides real-time data visualization, which is essential for the project's requirements.

Overall, this project has broadened my understanding of both database management and web application development, equipping me with valuable skills for future endeavors.