Willem Kroeger

CS 340: Client/Server Development

Southern New Hampshire University

June 17, 2023

**Animal Rescue Dashboard README**

This README file provides documentation for the Animal Rescue Dashboard project. It includes information on the required functionality of the project, the tools used, and the steps taken to complete the project. It also addresses any challenges encountered during the development process.

**Required Functionality**

The Animal Rescue Dashboard project aims to create a web application that allows users to view and filter animal rescue data. The required functionality includes:

1. Displaying a table of animal rescue data from the Austin Animal Center.
2. Allowing users to filter the data based on desired rescue type and (optionally) the desired breed.
3. Updating the table dynamically when filters are applied or removed.

This table shows the required breeds for each rescue type. When a rescue type is selected a database query is run, sorted by these fields within the data. Additional filtering is possible with the preferred breeds dropdown list on the dashboard to view only specific breeds.

A picture containing text, screenshot, font, number

Description automatically generated

**A screenshot of a computer

Description automatically generated with medium confidenceDashboard with no filters**

**A screenshot of a map

Description automatically generated with medium confidenceDisaster Rescue, sorted by golden retriever.**

**A screenshot of a computer

Description automatically generated with medium confidenceMountain Rescue, sorted by German Shepard mixes.**

**Water Rescue, sorted by Labrador Retriever mixes.**

A screenshot of a computer

Description automatically generated with medium confidence

**Tools Used**

The following tools were used to achieve the required functionality of the project:

**Python**: The programming language used for the backend development and data processing. Python was chosen as the programming language due to its versatility, ease of use, and extensive libraries for data processing and web development. <https://www.python.org/>

**Dash & Plotly**: A graphing library used for visualizations in the web application. Plotly was used for creating visualizations in the web application due to its rich and interactive graphing features.. Dash was selected as the web framework because it integrates well with Python and allows for the creation of interactive and responsive web applications. <https://dash.plotly.com/>

**Pandas**: A data manipulation library used for data filtering and processing. Pandas was utilized for data filtering and processing because of its powerful data manipulation capabilities. <https://pandas.pydata.org/>

**MongoDB**: A NoSQL database used as the model component for storing the animal rescue data. MongoDB was chosen as the model component for the project because it provides a flexible and scalable solution for storing and retrieving data. It integrates well with Python through libraries such as pymongo, allowing seamless interaction between the web application and the database. <https://www.mongodb.com/>

**Jupyter Notebooks**: Interactive Jupyter Notebooks were also used during the development of the project. Jupyter Notebooks provide an interactive computing environment that allows for easy prototyping, data exploration, and documentation. They enable the execution of code cells, viewing of results, and markdown cells for documenting code and project details. Jupyter Notebooks were used to develop and test code snippets, visualize data, and document the project. They played a crucial role in the iterative development process and collaborative work. <https://jupyter.org/>

**MongoDB and its Benefits**

MongoDB was used as the model component of the development for the following reasons:

Flexible Document Structure: MongoDB's document-based model allows for a flexible and schema-less data structure. This is beneficial when dealing with diverse and evolving data, such as animal rescue data, where different animals may have varying attributes.

Scalability: MongoDB is designed to scale horizontally, making it suitable for handling large amounts of data. As the animal rescue database grows, MongoDB can easily handle the increased data volume without sacrificing performance.

High Performance: MongoDB's indexing capabilities and query optimization make data retrieval efficient and fast. This is crucial for a web application that requires quick response times when filtering and displaying data.

Seamless Integration with Python: MongoDB provides a Python driver called pymongo, which simplifies the interaction between the Python backend and the database. This allows for easy data retrieval and manipulation within the web application.

**Dash Framework**

The Dash framework was used to provide the view and controller structure for the web application. Dash is the original low-code framework for rapidly building data apps in Python. It allows for the creation of data-driven dashboards with rich visualizations and interactive components.

Dash provides the following benefits for the Animal Rescue Dashboard project:

Easy Integration: Dash seamlessly integrates with Python, allowing developers to leverage their existing Python knowledge and libraries.

Interactive Components: Dash provides a wide range of interactive components, such as dropdowns, sliders, and graphs, which enhance the user experience and enable dynamic interactions with the data.

Responsive Design: Dash automatically handles the responsiveness of the web application, ensuring that it looks and functions well on different devices and screen sizes.

Reusability: Dash components and layouts are modular, making it easy to reuse and organize code components. This promotes code maintainability and extensibility.

**Reproducing the dashboard**

1. Install Python: Make sure Python is installed on your machine.
2. Install Jupyter Notebooks.
3. Install Required Libraries: You will need to install the required libraries to fully reproduce this work. The required libraries are: Dash, Pandas, Plotly, pymongo.

A screenshot of a computer program

Description automatically generated with medium confidence

1. Install MongoDB: Set up MongoDB on your machine.
2. Import the data into your MongoDB Database
3. Update the Configuration: In the python code, update the MongoDB connection details to match your database configuration.
4. Run the application and access the dashboard by connecting to the localhost settings inn your web browser.

**Development of the dashboard**

Developing this dashboard required understanding of the underlying dataset, specifically the nomenclature used for each breed. For example, when running a query searching for a Chesapeake Bay Retriever, it was discovered that the dataset had abbreviated it to “Chesa Bay Retr.” Each breed had to be located in the dataset to find a common search query for each breed that would display results from the dataset.

A screenshot of a computer code

Description automatically generated with low confidence

Additionally, the preferred breeds of dogs are typically listed as a mix of different dog breeds. Rather than removing these dogs that may be suitable for Grazioso Salvare to use, the breeds were searched for by using the text strings, so that even if they were a mix of two different breeds the dog would be listed.

**Challenges Encountered**

During the development of the Animal Rescue Dashboard, the following challenges were encountered:

Data Filtering: Implementing the dynamic data filtering based on rescue type and breed required careful handling of the filter logic and integrating it into the Dash application. It required a thorough understanding of the data structure and appropriate filtering logic to display the relevant data.

MongoDB Integration: Establishing a connection with MongoDB and retrieving data from the database using pymongo required understanding the MongoDB Python driver and the query syntax. It was necessary to ensure the correct configuration of the connection settings and handle potential errors during the connection process.

User Interface Design: Designing an intuitive and visually appealing user interface that effectively presented the animal rescue data required careful consideration of the layout, component placement, and styling. It was important to strike a balance between functionality and aesthetics.

These challenges were overcome by conducting thorough research, referring to official documentation and community resources, and leveraging knowledge and experience with Python, Dash, and MongoDB. Additionally, the community of support at <https://stackoverflow.com/> was invaluable in solving numerous issues and developing this dashboard.