Robert Cook

CS-340 Client/Server Development

Final Project README

April 21, 2024

**Functionality**

This program offers three primary functions. Firstly, it utilizes radial buttons to categorize data according to breed, age, and gender traits for various types of rescue dogs sought by Grazioso Salvare. Upon selecting any radial button, the application executes database queries and presents an updated data frame reflecting the desired criteria. The "Reset" radial button restores the table to its original, unfiltered state. The second function involves mapping and real-time updates. It initially displays a map pinpointing the location of the first entry in the data frame. As users choose rows, the map adjusts the marker to reflect the selected item. The third function generates a dynamic pie chart by sorting the data by breed and producing a chart based on the current data frame (excluding the entire database) showcasing the distribution of animal breeds.

**Dashboard Overview/Reset filters**A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Water Rescue Filter**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Disaster Rescue Filter**

A screenshot of a dashboard

Description automatically generatedA screenshot of a computer

Description automatically generated

**Mountain or Wilderness Rescue Filter**

A screenshot of a dashboard

Description automatically generated

A screenshot of a computer

Description automatically generated

**Tools**

The tools used are as follows:

* Python – Download Python from this link: https://realpython.com/installing-python/.
* Jupyter Notebooks - To install follow this link https://jupyter.org/install.
* MongoDB – Use this link to find download and installation instructions: https://docs.mongodb.com/manual/installation/.
* Plotly – Plotly needs to be imported to get correct charts for Python. Local copy of it can be found here: https://plotly.com/python/getting-started/
* Dash – This is used to build web applications. Install dash by following this link: https://pypi.org/project/dash/
* Pandas – This is used in the web application as well. This is used to create data frames in Python. Install it here: https://pandas.pydata.org/pandas-docs/stable/getting\_started/install.html

**Motivation**

Mongo was chosen for this program due to its ability to swiftly set up the database using a CSV document and its compatibility with Python. While Python can interact with SQL databases, transitioning between the syntax of SQL and Python can be inconvenient. The querying tools for executing basic CRUD functions in SQL databases tend to be more intricate compared to those in MongoDB when utilized with Python.

The choice of Dash for constructing the dashboard stemmed from its dynamic capabilities. Dash, built on React JavaScript, offers a highly responsive framework. It employs HTML Dash tags to manage outputs to different segments. Subsequently, updates to specified target inputs within the app are processed according to instructions outlined in the Python module.

**Steps Taken**

* **Database Configuration**: Implemented CRUD operations in Python to interact with MongoDB.
* **Dashboard Creation**:

• Constructed an interactive data table showcasing shelter animal information.

• Designed filter functionalities catering to Water Rescue, Mountain or Wilderness Rescue, and Disaster Rescue.

• Integrated widgets for dynamic data visualization, such as a geolocation chart and supplementary chart options.

* **Testing and Deployment**:

• Conducted thorough testing to verify the dashboard's functionality.

• Locally deployed the dashboard to facilitate user interaction.

• Generated screenshots illustrating different states of the dashboard for demonstration purposes.

**Challenges**

Some of the challenges I faced were the implementation of the dynamic pie chart to display different values according to the filter selected. The next was creating a single function to update the dashboard and map and not create duplicate callback errors. I did not seem to run into too many challenges along the way just small road bumps.