7-2 Project Two Submission: README file

Natalia A. Santiago

Southern New Hampshire University

CS-340: Client/Server Development

Tarik Iles

December 9, 2024

Grazioso Salvare Dashboard Project

This project entails developing a web application dashboard for the innovative International Rescue-animal training company, the Grazioso Salvare. The dashboard connects to MongoDB to present a Dynamic/Interactive user experience by letting the student manage / visualize data originating from animal shelters around Austin Texas. The objective is to identify / categorize dogs that are suitable for different types of search-and-rescue training.

**Requirements**

The dashboard provides users with interaction through the following capabilities with the web application.

1. Interactive Filtering Options: Give users the capability to filter the data on the type of rescue, such as Water Rescue, Mountain/Wilderness Rescue, and Disaster/Individual Tracking. The provided filtering options provide a reset option for going back to the original data.

A screenshot of a computer

Description automatically generated

1. Dynamic Data Table: Updates interactively along with the filtering options and displays the relevant data concerning the MongoDB database.A screenshot of a computer

   Description automatically generatedA screenshot of a computer

   Description automatically generatedA screenshot of a computer

   Description automatically generated
2. Visualization Charts: Pie chart on breed distribution and a geolocation map on animal locations dynamically updated based on the selection of filters.

A screenshot of a computer

Description automatically generated

1. Branding Elements: Include the Grazioso Salvare logo linked to their homepage, as well as a unique identifier to credit the developer.

A screenshot of a computer

Description automatically generated

**Tools and Technologies Used**

* MongoDB: NoSQL database used during development for the storage and management of data for animal shelters. Its flexible schema and advanced querying features make it highly suitable for maintaining a large quantum of varied information on rescue animals.
* Python: The Python language is utilized to drive all the back-end CRUD operations and interacts with MongoDB.
* Dash by Plotly: A Python framework used in developing the dashboard for the interactive web app. The ease of use and the ability to create complex, user-friendly web applications with Dash using a few lines of Python code justified simplifying it.
* Dash Leaflet: An extension of Dash used to create the geolocation map.
* Plotly Express: A high-level wrapper for Plotly used to develop the interactive pie chart.

**Development and Rationale**

**MongoDB for Model Component**

The reason behind picking MongoDB is that its flexible, schema-less data structure goes well with the variable and dynamic data taken from animal shelters. The strong query capabilities allow MongoDB to efficiently retrieve and manipulate data. The integration of MongoDB in Python through libraries such as pymongo makes MongoDB ideal to be used in this project.

**Dash Framework**

The application of choice is Dash for the ease with which one can build web applications that are interactive and data-driven using pure Python. In summary, Dash abstracts much of the complexity in developing web applications by allowing concentration on the core functionalities being developed within the application. It also has easy integrations with Plotly for interactive charts and Dash Leaflet for geolocation maps.

**Project Steps**

1. Setup and Configuration: Installation of essential Python packages, followed by the setup of the development environment.
2. Integrate Database: Designed a CRUD module for MongoDB to interact with it and perform the common create, read, update, and delete operations.
3. Dashboard Layout: It designs the layout of the dashboard using data table, filtering options, and visualization charts through Dash components.
4. Interactive Features: Implementation of callback functions in Dash to refresh data table and charts dynamically based on input provided.
5. Testing and Deployment: Prolonged testing to ensure everything works just fine; went ahead with the deployment of the dashboard.

**Challenges and Solutions**

* Data Conversion Errors: Issues with string values to numeric conversions for a geolocation map. Added solution: validation checks through handling exceptions to make sure only valid numerical data is utilized.
* Deprecation Warnings: Received deprecation warnings emanating from some of the Dash packages. The solution came in updating the import statements to the recommended new versions.

**References**

*3.13.0 Documentation*. (2024). Python.org. <https://docs.python.org/3/?form=MG0AV3>

*Dash Documentation & User Guide | Plotly*. (2024). Plotly.com. <https://dash.plotly.com/?form=MG0AV3>

*MongoDB Documentation*. (2024). Mongodb.com. <https://docs.mongodb.com/?form=MG0AV3>

*Plotly*. (2024). Plotly.com. <https://plotly.com/python/?form=MG0AV3>