7-2 Project Two

Debynyhan Banks

CS 340 Client/Server Development

Southern New Hampshire University

Professor Paruchuri

April 16, 2023

**Project Overview**

Grazioso Salvare, a business that specializes in animal rescue, commissioned the development of the Animal Shelter Dashboard, an interactive web application. Users can filter and visualize the animal rescue type using various criteria such as water rescue, disaster or individual tracking, and mountain or wilderness. Additionally, there is reset that will return all widgets to their original unfiltered state. Information about the animals kept in the shelter is displayed on the dashboard. Grazioso Salvare must make wise choices when it comes to animal rescue missions. To make these decisions, they must have access to the most up-to-date information. Through Grazioso Salvare's dashboard, they have access to the information they need, enabling them to make decisions quickly and efficiently.

**Required Functionality**

* Retrieve data from a MongoDB database.
* Display a table with shelter information about animals.
* Enable users to filter the data based on predefined criteria (disaster, water, mountain, reset) using radio buttons.
* Display a pie chart to show how the filtered data's breed distribution is distributed.
* Show a map with the first dog’s location from the filtered data.
* Unfiltered view of the dataset.

**Tools Used**

* [PyMongo](https://www.mongodb.com/docs/drivers/pymongo/#pymongo)
* [MongoDB](https://www.mongodb.com/)
* [Python](https://www.python.org/)
* [Jupyter](https://jupyter.org/)
* [Dash](https://plotly.com/?_gl=1*1rezpkw*_ga*MTY3NjA3MTg1NC4xNjgwMDEyNjg5*_ga_6G7EE0JNSC*MTY4MTMxMTQwNy4xMi4xLjE2ODEzMTIwNDguMC4wLjA.)
* [Plotly](https://plotly.com/?_gl=1*k51o3a*_ga*MTY3NjA3MTg1NC4xNjgwMDEyNjg5*_ga_6G7EE0JNSC*MTY4MTMxMTQwNy4xMi4xLjE2ODEzMTE0NTAuMC4wLjA.)
* [Dash Leaflet](https://pypi.org/project/dash-leaflet/)

**Rationale for why these tools were used**

The official Python driver for MongoDB is [PyMongo](https://www.mongodb.com/docs/drivers/pymongo/#pymongo). The connection between the Python program and the MongoDB database was made easier with its help. PyMongo offers a straightforward and easy-to-use interface for Python data manipulation and querying in MongoDB. All of MongoDB's CRUD (Create, Read, Update, Delete) operations are supported by PyMongo, and it also offers a simple way to build indexes. PyMongo is also extremely scalable and suitable for distributed systems.

[MongoDB](https://www.mongodb.com/) is a powerful, flexible, and scalable NoSQL database. It was chosen because it can store a lot of data in a flexible, JSON-like format (BSON) which is simple to integrate with Python. MongoDB offers an effective means of data manipulation and querying and is particularly well suited for handling intricate data structures and relationships.

[Python](https://www.python.org/) was chosen as the main programming language for the project because of its adaptability, wide library support, and acceptance in the data science and web development communities. Python is a great option for tasks like this dashboard because it is simple to read and understand.

[Plotly](https://plotly.com/?_gl=1*k51o3a*_ga*MTY3NjA3MTg1NC4xNjgwMDEyNjg5*_ga_6G7EE0JNSC*MTY4MTMxMTQwNy4xMi4xLjE2ODEzMTE0NTAuMC4wLjA.) was chosen because it is a powerful graphing library for Python that provides a wide range of interactive chart types. A pie chart visualization was created within the dashboard using Plotly. Dash integrates seamlessly with Plotly, making it the ideal tool for creating interactive visualizations.

[Dash Leaflet](https://pypi.org/project/dash-leaflet/) is a lightweight mapping library for creating interactive maps in Dash applications. The dashboard's map was used to show the locations of the animals. Dash Leaflet, which is based on the well-known Leaflet.js library, offers a straightforward but effective way to integrate interactive maps into Dash applications. It is simple to use and lets you style the maps however you like. In addition to being effective and lightweight, it is the ideal option for displaying maps in Dash applications.

[Jupyter](https://jupyter.org/) is an open-source web application for creating and sharing documents that contain live code, equations, and visualizations. Jupyter was used during the development process for testing and debugging the code.

**Reproducing the Project**

Jupyter was used to develop and test the Gravioso Savalare’s Dash interactive web application. Ensure that you have Jupyter installed before proceeding, you can download [Jupyter](https://jupyter.org/) at their official website.

## **Installation**

The steps and tools for installing MongoDB locally on a Mac M1 are covered in this installation guide.

**Install Jupyter**

You can write, run, and analyze code in real time using Jupyter Notebook, an interactive environment. This feature streamlines the development and debugging process by enabling you to test your CRUD operations and MongoDB queries right away.

* Install the Jupyter Notebook using the official Jupyter website.

**Install Homebrew**

Homebrew is a package manager for macOS that simplifies the process of installing, updating, and managing software. The Homebrew **brew** package is not included with macOS by default.

* Install brew using the official [Homebrew installation instructions](https://brew.sh/#install).

**Install Homebrew XCode command line tools**

The Xcode command-line tools from Apple's Xcode are necessary for Homebrew.

* Install the Xcode command-line tools by running the following command in your macOS Terminal: “xcode-select –install"

**Installing MongoDB 4.2 Community Edition**

* Install the “MongoDB Community 4.2 Edition” using the official [Install MongoDB Community Edition](https://www.mongodb.com/docs/v4.2/administration/install-community/)

### **Run MongoDB Community Edition**

Run the following command in a terminal to make MongoDB (also known as the [mongod](https://www.mongodb.com/docs/v4.2/reference/program/mongod/#bin.mongod) process) available as a macOS service:

* brew services start mongodb-community

*Note: To use the MongoClient localhost, copy the port number*.

To terminate a [mongod](https://www.mongodb.com/docs/v4.2/reference/program/mongod/#bin.mongod) that is running as a macOS service, issue the following command:

* brew services stop mongodb-community

### **Connect and Use MongoDB**

To begin using MongoDB, connect a [mongo](https://www.mongodb.com/docs/v4.2/reference/program/mongo/#bin.mongo) shell to the running instance. From a new terminal, issue the following command: “mongo”

**Download the AAC Outcomes data set:**

[Austin Animal Center Outcomes Spreadsheet](https://learn.snhu.edu/content/enforced/1272220-CS-340-T4220-OL-TRAD-UG.23EW4/course_documents/aac_shelter_outcomes.csv?_&d2lSessionVal=8VtJTKNqXYE5enHWGNVkWGt2q&ou=1272220)

**Import the data set into MongoDB:**

Use the mongoimport tool to import the AAC Outcomes data set into a MongoDB database named "AAC" and a collection named "animals". You can do this by running the following command in your terminal: `mongoimport --port 48356 --db AAC --collection animals --type csv --headerline --file =./aac\_shelter\_outcomes.csv.

Install the necessary Python libraries using pip: `**pip install pymongo dash plotly dash-leaflet jupyter`**

**Clone the repository**

[**https://github.com/Debynyhan/Dash-Dashboard.git**](https://github.com/Debynyhan/Dash-Dashboard.git)

**Challenges and Solutions**

Understanding the structure and usage of the Dash framework was a challenge that I had to overcome. I overcame this obstacle by studying Dash documentation and examples to learn how to create interactive components and callback functions. Another challenge I faced was filtering data based on user selections and updating the dashboard components accordingly. However, I created callback functions that listen to changes in user input and update the data table, pie chart, and map components dynamically. Another challenge I faced was integrating the animal data from MongoDB with the Dash app. I resolved this issue by reading the Pandas documentation. I used the Pandas library to convert the MongoDB data into a DataFrame, which can be easily manipulated and displayed in the Dash components.