**Project Description**

This project is designed to develop a user-friendly interface for Grazioso Salvare, an innovative animal rescue training company, to streamline and simplify the management of their animal rescue database. The goal is to create an interactive dashboard that allows users to access and analyze data related to animal rescues, helping Grazioso Salvare efficiently manage their animal adoption processes.

**Required Functionality**

The required functionality of the project includes:

1. Interactive Dashboard: Develop an interactive web-based dashboard that allows users to access and analyze data from the animal rescue database.
2. Data Filtering: Implement filtering options, such as radio buttons and dropdowns, to filter and display data based on rescue types, including Water Rescue, Mountain, Wilderness, Disaster Rescue, and Individual Tracking.
3. Data Visualization: Include data visualization components, such as charts and graphs, to provide insights into the distribution of rescue types.
4. Reset Filters: Add a "Reset Filters" button that allows users to clear all filters and view the complete dataset.
5. Connection to MongoDB: Establish a connection to the MongoDB database to fetch and display data.
6. User Interface: Create a user-friendly and visually appealing interface, including a logo (Grazioso Salvare's logo) and a well-organized layout.

**Tools Used**

To achieve the required functionality, the following tools and technologies were used:

* Python: Python was chosen as the primary programming language for its versatility and extensive libraries, making it suitable for data manipulation, web development, and connecting to MongoDB.
* Jupyter Notebook: Jupyter Notebook was used for developing and running the Python code. It provides an interactive and easy-to-use environment for data analysis and visualization.
* Dash by Plotly: Dash is a Python web application framework for building interactive web-based data dashboards. It was used to create the dashboard's view and controller components.
* MongoDB: MongoDB was chosen as the model component of the development because of its NoSQL nature, flexibility, and scalability. It provides an efficient way to store and retrieve unstructured or semi-structured data, making it suitable for storing animal rescue data.
* PyMongo: PyMongo, a Python driver for MongoDB, was used to establish a connection between Python and the MongoDB database. It allows for seamless interaction with the database and data retrieval.

**MongoDB Usage**

MongoDB was selected as the database model for several reasons:

* Flexible Schema: MongoDB's document-based structure allows for flexible and dynamic data storage, which is ideal for managing animal rescue data where the attributes may vary.
* Scalability: MongoDB can handle large volumes of data and scale horizontally as the dataset grows, making it suitable for accommodating Grazioso Salvare's expanding animal rescue database.
* JSON-Like Documents: MongoDB stores data in BSON (Binary JSON) format, which aligns well with Python's native data structures like dictionaries and lists, making it convenient for Python integration.
* Efficient Queries: MongoDB provides efficient querying capabilities, which are crucial for retrieving specific data subsets based on various filtering options.

**Dash Framework**

The Dash framework was used to create the web-based dashboard:

* Interactivity: Dash allows for the creation of interactive web applications with components like radio buttons, dropdowns, and buttons, enabling users to filter and manipulate data.
* Data Visualization: Dash seamlessly integrates with Plotly, which provides a wide range of data visualization options, including charts, graphs, and maps.
* Pythonic: Dash is written in Python and utilizes Python syntax, making it accessible to Python developers with minimal additional learning.
* Customization: Dash offers a high degree of customization for the user interface, enabling the incorporation of Grazioso Salvare's branding and logo.

**Steps to Complete the Project**

The following steps were taken to complete the project:

1. Environment Setup: Ensure that Python, Jupyter Notebook, PyMongo, Dash, and other necessary libraries are installed on the local machine.
2. Database Connection: Establish a connection to the MongoDB database hosted at host:port using the provided credentials (username and password).
3. Data Retrieval: Fetch the animal rescue data from the MongoDB database using the AnimalShelter class and store it in a Pandas DataFrame.
4. Dashboard Development: Create a Jupyter Notebook (ProjectTwoDashboard.ipynb) for developing the dashboard. Use Dash to design the user interface, implement filtering options, and integrate data visualization components.
5. Interactive Filters: Implement interactive filtering options, including radio buttons and dropdowns, to filter data based on rescue types.
6. Data Visualization: Create interactive data visualizations, such as pie charts, to showcase the distribution of rescue types within the dataset.
7. Reset Filters: Develop a "Reset Filters" button that allows users to clear all filters and view the complete dataset.
8. User Interface: Customize the user interface by incorporating Grazioso Salvare's logo and ensuring a visually appealing layout.
9. Testing: Thoroughly test the dashboard's functionality to ensure it meets the required specifications.
10. Documentation: Update the project README to provide clear instructions, explanations, and any necessary documentation.

**Challenges**

The largest challenge that I encountered was the same one that I have been struggling with for the entire term. For some reason no matter what I tried I could not get my code to run because it would give me the ModuleNotFoundError. Even with everything located within the same directory and triple checking all of my code I could not find where I made the mistake that caused this. This was the same error that I had in all of my previous assignments as well.

A screenshot of a computer

Description automatically generated

Because of this error, I was not able to run my program and see if what I did was working. To try to work around this challenge I tried to keep my code as simple as possible to limit the chance that I would make an error and I leaned heavily on the code that I already knew and the code that we had learned throughout this class. By doing this I am hoping that I have been able to write code that hopefully meets the requirements for this project.