# CS 340 Project Two README

## About the Project

This project uses a database and CRUD Python module to create a fully functional MongoDB dashboard. This dashboard will be used by the rescue-animal training company, Grazioso Salvare, to identify dogs that are good candidates for search-and-rescue training. The company requires an interactive data table that utilizes filtering options as well as a pie chart and geolocation chart to display the data. This allows the company to easily search for dogs within the animal shelter database that meet their requirements for search-and-rescue training.

The interactive data table I created allows the company to filter any of the data using user input, but more importantly there are dedicated buttons to quickly filter the database for dogs meeting water rescue, mountain or wilderness rescue, and disaster rescue or individual tracking criteria. The table allows the user to select an item, filter data, reset the filter, and move between pages. The pie chart displays the data within the table at the given time, and the geolocation chart displays the location of the first animal listed in the table at the given time.

Below are screenshots demonstrating the functionality of my dashboard, including the header name, Grazioso Salvare logo, data table, pie chart, and geolocation chart. The screenshots demonstrate the changes made after each search-and-rescue filter is set, as well as the reset filter.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, chart, application

Description automatically generated

A screenshot of a computer

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

A screenshot of a computer

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

## Installation

This project utilizes MongoDB and Python together to create and hold an extensive database document that is then utilized for complex queries through the interactive dashboard interface. MongoDB was chosen for this project because of its ease of use and ability to hold and configure a large amount of data. Python was chosen to interface with MongoDB through the PyMongo driver because of its compatibility and flexibility that makes it quick and simple to query for data and create a dashboard to display the data. The Dash framework was also used for this project to provide a simplified and organized method to develop the dashboard utilizing the database information.

Here are some useful links that explain the tools used for this project:

* <https://www.tutorialspoint.com/python_web_development_libraries/python_web_development_libraries_dash_framework.htm>
* <https://www.w3schools.com/python/python_mongodb_getstarted.asp>
* https://dash.plotly.com/dash-core-components

## Steps Taken

To get this project running locally there are a few simple steps to follow. First, the CSV file containing the database information must be imported into MongoDB using a database called AAC with a collection called Animals. User authentication must then be set up in order to get access to the database to create and read files within it. The Python module containing the create, read, update, and delete functions of CRUD was then created in order to provide interaction with the database. The create method inserts a document into the database collection, the read function queries for documents from the collection within the database, the update function updates existing items, and the delete function removes items from the database. I did run into a small issue when testing the Python file; the results of the create method test returned blank instead of true. In order to fix this, MongoDB needs to be actively running on the machine in order for a successful connection to occur between the Python file and MongoDB. The next step is to create a IPYNB file that imports the PY file already created. Within this new file, the dashboard for the project is created using the Dash framework.

## Challenges

I faced very few challenges while creating this project, as the tools used are meant to be simple and quick to implement. Overall, a solid understanding of MongoDB, Python, and the Dash framework are necessary in order to complete this project. Some troubleshooting may be required throughout development, as each component of this project relies on the others because they are all intertwined.

## Contact

Miranda Putnam